

0063389



TECH LIBRARY KAFB, NM

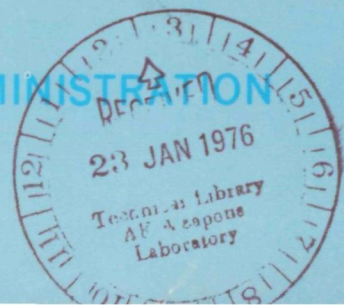
# APPLICATIONS OF COMPUTER GRAPHICS IN ENGINEERING

A conference held at  
LANGLEY RESEARCH CENTER  
Hampton, Virginia  
October 1-2, 1975



LOAN COPY: RETURN TO  
AFWL TECHNICAL LIBRARY  
KIRTLAND AFB, N. M.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION







# APPLICATIONS OF COMPUTER GRAPHICS IN ENGINEERING

*A conference held at Langley Research Center  
Hampton, Va., October 1-2, 1975*



*Scientific and Technical Information Office* 1975  
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION  
Washington, D.C.



For sale by the National Technical Information Service  
Springfield, Virginia 22161  
Price — \$15.25



## PREFACE

For a number of years computer graphics has been emerging as a tool to aid in the interpretation, evaluation, and dissemination of information generated for and by computers. The present interest in computer graphics has been well demonstrated by the large attendance at two conferences on computer graphics sponsored by the Special Interest Group on Graphics of the Association of Computing Machinery. The first of these conferences, held in July 1974 at the University of Colorado, drew over 500 attendees. Many excellent papers were presented which discussed relevant computer science issues for both passive and interactive graphics. The increased emphasis on computer graphics manifested at that conference was due in part to the widespread usage of time-share operating systems and the emergence of low-cost graphic CRT terminals of both the refreshed and storage type. At that time the organizing committee of the present conference concluded that a forum was needed to bring together the computer science and application issues so the new tool, computer graphics, could be more efficiently applied to problems facing the engineering community. This has led to the present conference on Applications of Computer Graphics in Engineering. The papers presented together with the panel discussion are included herein and describe the application of computer graphics in a variety of engineering disciplines. It is hoped that the information presented will stimulate engineers to apply computer graphics to the solution of their problems.

The conference program was planned under the direction of a conference steering committee composed of

David D. Loendorf, Coordinator	Langley Directorate - USAAMRDL
Mary S. Adams	NASA, Langley Research Center
John P. Decker	NASA, Langley Research Center
Robert E. Fulton	NASA, Langley Research Center
Roy V. Harris, Jr.	NASA, Langley Research Center
James M. Ortega	Institute for Computer Applications in Science and Engineering
Robert E. Smith, Jr.	NASA, Langley Research Center
Robert G. Voigt	Institute for Computer Applications in Science and Engineering



**Page Intentionally Left Blank**



## CONTENTS

PREFACE . . . . .	iii
-------------------	-----

### ANALYSIS AND DESIGN I

1. DEVELOPMENT OF AN ENGINEERING COMPUTER GRAPHIC LABORATORY . . . . .	1
David C. Anderson and Richard E. Garrett (Purdue University)	
2. AUTOMATED FINITE ELEMENT GRID BREAK-UP METHOD - A VERIFICATION OF THE SIX NODE AVERAGING APPROACH . . . . .	15
H. J. Barten (Pratt & Whitney Aircraft)	
3. APPLICATION OF THE CINGEN PROGRAM - A THERMAL NETWORK DATA GENERATOR . . . . .	27
William E. Schultz and Ronald D. Schmitz (Sperry Support Services)	
4. QUICK-GEOMETRY - A RAPID RESPONSE METHOD FOR MATHEMATICALLY MODELING CONFIGURATION GEOMETRY . . . . .	49
Alfred F. Vachris, Jr., and Larry S. Yaeger (Grumman Aerospace Corp.)	

### ANALYSIS AND DESIGN II

5. COMPUTER-GENERATED ANIMATION FOR ANALYSIS AND DESIGN . . . . .	75
Larry J. Feeser (Rensselaer Polytechnic Institute)	
6. THE CONTROL DATA "GIRAFFE" SYSTEM FOR INTERACTIVE GRAPHIC FINITE ELEMENT ANALYSIS . . . . .	83
S. Park and D. M. Brandon, Jr. (Control Data Corporation)	
7. THREE-DIMENSIONAL STRUCTURAL ANALYSIS USING INTERACTIVE GRAPHICS . . .	111
Johnny H. Biffle and Hugh A. Sumlin (Sandia Laboratories)	
8. GRAPHICS FLUTTER ANALYSIS METHODS - AN INTERACTIVE COMPUTING SYSTEM AT LOCKHEED-CALIFORNIA COMPANY . . . . .	133
Nick A. Radovcich (Lockheed-California Company)	

### ANALYSIS AND DESIGN III

9. INTERACTIVE COMPUTER AIDED TECHNOLOGY: EVOLUTION IN THE DESIGN/MANUFACTURING PROCESS . . . . .	167
C. H. English (McDonnell Aircraft Company)	

10. APPLICATIONS OF COMPUTER GRAPHICS TO AIRCRAFT SYNTHESIS . . . . .	189
Ralph L. Carmichael and Richard Putnam (NASA Ames Research Center)	
11. USE OF GRAPHICS IN THE DESIGN OFFICE AT THE MILITARY AIRCRAFT DIVISION OF THE BRITISH AIRCRAFT CORPORATION . . . . .	203
W. A. Coles (British Aircraft Corporation Ltd.)	
12. INTERACTIVE COMPUTER GRAPHICS SYSTEM FOR STRUCTURAL SIZING AND ANALYSIS OF AIRCRAFT STRUCTURES . . . . .	233
Dror Bendavid, Aharon Pipano, Alex Raibstein, and Emil Somekh (Israel Aircraft Industries Ltd.)	
13. COMPUTER GRAPHICS APPLICATION IN THE ENGINEERING DESIGN INTEGRATION SYSTEM . . . . .	257
C. R. Glatt, R. W. Abel, G. N. Hirsch, G. E. Alford, W. N. Colquitt, and W. A. Stewart (Sigma Corporation)	

#### ENGINEERING RESEARCH

14. SOME RESEARCH ADVANCES IN COMPUTER GRAPHICS THAT WILL ENHANCE APPLICATIONS TO ENGINEERING DESIGN . . . . .	287
John J. Allan III (University of Texas)	
15. AN INTERACTIVE GRAPHICS PROGRAM TO RETRIEVE, DISPLAY, COMPARE, MANIPULATE, CURVE FIT, DIFFERENCE AND CROSS PLOT WIND TUNNEL DATA . . . . .	297
Robert D. Elliott, Norbert M. Werner, and William M. Baker (Lockheed California Company)	
16. APPLICATION OF INTERACTIVE COMPUTER GRAPHICS IN WIND-TUNNEL DYNAMIC MODEL TESTING . . . . .	325
Robert V. Doggett, Jr., and Charles E. Hammond (NASA Langley Research Center)	

#### SIMULATION

17. COMPUTER GRAPHICS IN ARCHITECTURE AND ENGINEERING . . . . .	355
Donald P. Greenberg (Cornell University)	
18. COCKPIT DESIGN AND EVALUATION USING INTERACTIVE GRAPHICS . . . . .	361
Susan M. Evans (University of Dayton Research Institute)	
19. AN IMPROVED HUMAN DISPLAY MODEL FOR OCCUPANT CRASH SIMULATION PROGRAMS . . . . .	375
K. D. Willmert (Clarkson College of Technology) and T. E. Potter (Xerox Corporation)	

20. A FLEXIBLE FLIGHT DISPLAY RESEARCH SYSTEM USING A GROUND-BASED  
INTERACTIVE GRAPHICS TERMINAL . . . . . 387  
Jack J. Hatfield, Henry C. Elkins, Vernon M. Batson,  
and William L. Poole (NASA Langley Research Center)
21. THE APPLICATION OF INTERACTIVE GRAPHICS TO LARGE TIME-DEPENDENT  
HYDRODYNAMICS PROBLEMS . . . . . 419  
Fred Gama-Lobo and Lynn D. Maas (Los Alamos Scientific  
Laboratory)

#### ENGINEERING INTERFACE

22. COMPUTER GRAPHICS FOR MANAGEMENT - AN ABSTRACT OF CAPABILITIES AND  
APPLICATIONS OF THE EIS SYSTEM . . . . . 427  
Barry J. Solem (Boeing Computer Services, Inc.)
23. SURFACE FITTING THREE-DIMENSIONAL BODIES . . . . . 447  
Fred R. DeJarnette and C. Phillip Ford III  
(North Carolina State University)
24. ENGINEERING COMPUTER GRAPHICS IN GAS TURBINE ENGINE DESIGN,  
ANALYSIS, AND MANUFACTURE . . . . . 475  
Richard S. Lopatka (Pratt & Whitney Aircraft)
25. PACKAGING PRINTED CIRCUIT BOARDS - A PRODUCTION APPLICATION OF  
INTERACTIVE GRAPHICS . . . . . 495  
Walter A. Perrill (General Dynamics)
26. DESIGN AND IMPLEMENTATION OF A MEDIUM SPEED COMMUNICATIONS INTERFACE  
AND PROTOCOL FOR A LOW COST, REFRESHED DISPLAY COMPUTER . . . . . 521  
James R. Rhyne and Mart D. Nelson (The University of Houston)
27. BIG SYSTEM - INTERACTIVE GRAPHICS FOR THE ENGINEER . . . . . 535  
Charles E. Quenneville (Boeing Computer Services, Inc.)

#### POSTER SESSION

28. A GRAPHICS APPROACH IN THE DESIGN OF THE DUAL AIR DENSITY  
EXPLORER SATELLITES . . . . . 555  
David S. McDougal (NASA Langley Research Center)
29. A COMPUTER PROGRAM FOR FITTING SMOOTH SURFACES TO THREE-DIMENSIONAL  
AIRCRAFT CONFIGURATIONS . . . . . 569  
Charlotte B. Craidon and Robert E. Smith, Jr.  
(NASA Langley Research Center)



30. MESH GENERATION FOR TWO-DIMENSIONAL REGIONS USING THE TEKTRONIX DVST  
(DIRECT VIEW STORAGE TUBE) GRAPHICS TERMINAL . . . . . 587  
Verlan K. Gabrielson (Sandia Laboratories)
31. AN INTERACTIVE GRAPHICS PACKAGE FOR THE AUTOMATIC NODE RENUMBERING  
OF FINITE ELEMENT MATRICES . . . . . 605  
Ronald F. Boisvert and William G. Poole, Jr.  
(College of William and Mary and ICASE)
32. SPACEBAR KINEMATIC DESIGN BY COMPUTER GRAPHICS . . . . . 615  
Richard J. Ricci (Lockheed-California Company)
33. SIKORSKY INTERACTIVE GRAPHICS SURFACE DESIGN/MANUFACTURING SYSTEM . . . 633  
Richard Robbins (Sikorsky Aircraft)
34. THE DESIGN AND IMPLEMENTATION OF CRT DISPLAYS IN THE  
TCV REAL-TIME SIMULATION . . . . . 641  
John B. Leavitt and Syed I. Tariq (Electronic Associates, Inc.);  
and George G. Steinmetz (NASA Langley Research Center)

PRACTICAL HARDWARE, SOFTWARE, AND SYSTEM CONSIDERATIONS

35. PANEL DISCUSSION . . . . . 653  
S. H. Chasen, Moderator (Lockheed-Georgia Company),  
C. Machover (Information Display, Inc.),  
J. Foley (University of North Carolina),  
R. Phillips (University of Michigan), and  
H. Selling (Ford Motor Company)

# THE DEVELOPMENT OF AN ENGINEERING COMPUTER GRAPHICS LABORATORY

David C. Anderson  
School of Mechanical Engineering  
Department of Computer Science  
Purdue University

Richard E. Garrett  
School of Mechanical Engineering  
Purdue University

## INTRODUCTION

Computer graphics has rapidly gained recognition in industry and academia as a powerful tool for engineering design and analysis. This apparent popularity has generated increasing interest in new display hardware and an even greater demand for systems and applications software. In this atmosphere, the Computer Aided Design and Graphics Laboratory in the School of Mechanical Engineering at Purdue University evolved as an environment for research and education in interactive computer graphics.

This presentation describes the development of the laboratory in terms of the hardware and software system and several of the ongoing application-oriented projects, educational graphics programs, and graduate research projects. In keeping with the spirit of graphical communication, a motion picture film was prepared to graphically describe the laboratory and the projects. The remainder of this paper summarizes the material in this film, and the narration at the presentation.

## THE SYSTEM

The equipment configuration used by the majority of laboratory projects is depicted in figure 1. Current development efforts in interactive graphics employ the Digital Equipment Corporation PDP 11/40 minicomputer as the primary computation processor and the Imlac PDS-1 as an intelligent display processor. Data paths from the PDP to the Tektronix 4014 storage tube terminal, the Calcomp 502 76 cm (30 in.) flatbed plotter (via a microcomputer buffer processor), and the Purdue remote access terminal system provide a broad base for applications programming.

The software system consists of a FORTRAN IV subroutine package in the PDP 11/40 that interacts with a graphics operating system in the Imlac. Subroutine calls from the user's program executing in the PDP are transferred to the Imlac executive, where the corresponding routine is processed. The package comprises a comprehensive set of graphics routines for dynamic, structured two-dimensional display manipulation, and numerous routines to handle a variety of input devices at the Imlac.

The design objective during the continuing development of the system is to provide the application programmer (typically an engineer versed in FORTRAN) an interface between his problem and graphical interaction in as efficient and "learnable" manner as possible. The success of the interface, and consequently, the success of the system is measured by the applications it supports.

## APPLICATIONS

The following paragraphs briefly describe a representative sampling of engineering application projects conducted in the laboratory. The individual projects vary from small educational programs demonstrating solutions to engineering problems to graduate research in three-dimensional graphical systems. Figures 2 through 10 are single frame "snapshot" plots from each project, generated on the Calcomp 502 directly from the Imlac. Figure 1 was created interactively using a general purpose documentation program and plotted directly from the PDP 11/40.

The first project involves three dimensional data generation and manipulation. A three dimensional cursor controlled by the user manipulates objects consisting of parametric curves and surfaces. The generated data can be passed to analysis programs, or used as input to this program during subsequent design sessions. Figure 3 shows a pictorial view of a boat hull composed of six bi-cubic surface patches. The bottom view depicts the boundary curves shown in the top view with additional trace curves for greater surface detail.

The general transformation capabilities of the preceding system are demonstrated with the rotation of a three-dimensional linkage (figure 4). Real time rotation aids kinematics students in the visualization of complex three-dimensional motion.

Undergraduate students in Mechanical Engineering design courses "see" solutions to their design projects using the demonstration programs shown in figures 5 and 6. On-line modification of system parameters gives immediate analysis feedback to the student.

The results of a computer simulation of the dynamic motion of a reed valve compressor are displayed in figure 7. The program operates on the data produced by a large analysis program which is a research project conducted in the School of Mechanical Engineering by Dr. Werner Soedel and Dennis Strader. The display is interactively controlled by the user; thus graphical evaluation of the complex dynamical simulation is facilitated.

A program to display the vibration of a two degree of freedom spring-mass-damper system is shown in figure 8. The instructions beneath the graph describe the options available to the user. The program was a student project for the advanced computer graphics course.

Free transverse vibration of a cylinder fixed at both ends is displayed in figure 9. The user can start and stop the vibration, alter the modes of vibration, vary the linework grid size, and control the viewing direction interactively at the Imlac. For relatively coarse grids such as one shown in the figure,

the response rate is sufficient to view the motion as it is calculated. To compensate for the displeasingly slow response rate with large numbers of grid lines, a motion picture camera was mounted to the Imlac and cycled frame-by-frame by the PDP program. The film gives a smooth, realistic display of complex cylinder motion that in the past was virtually indescribable to mechanical vibration students.

The ability to produce high quality, realistic graphical output using the hidden line removal and shading programs is a valuable asset to the computer graphics laboratory. Figure 10 shows a view through the windows into the new laboratory site and a bow view of a sailboat hull. The block letter titles in figure 2 are actually a collection of three-dimensional planes that were automatically generated and passed to the hidden line program by a special lettering program. The data for these pictures can also be passed to the shading program that generates a continuously shaded image consisting of raster point data. This data is plotted on the Imlac or Tektronix screens to give a realistic representation of the original three-dimensional picture.

#### CONCLUDING REMARKS

The authors extend their appreciation to Michael Bailey, John Brewer, Donald Riley, and Dr. Phillip White for the use of their projects and time in preparing the film.

The following list of Purdue University theses document the development of the laboratory through a history of graduate research efforts:

1. Anderson, David C.  
"The Development of a General Purpose Interactive Graphics System", Ph.D., December 1974.
2. Applegate, Sheldon L.  
"An Interactive Computer Graphics Approach to Four Bar Linkage Design", M.S.M.E., August 1975.
3. Belleville, Robert L.  
"The Design and Development of an Interactive Computer Graphics System", M.S., June 1969.  
  
"Man-Machine Communication: An Examination of the Machines", Ph.D., August 1974.
4. Gunn, Moira A.  
"On the Development of Computer Graphic Design Tools for the Enhancement of Creativity", Ph.D., May 1974.
5. Palmer, John L., Lieutenant Colonel  
"Design Education Stimulated by Interactive Graphics Notation (DESIGN)", Ph.D., December 1973.

6. Putnam, Richard E.  
"A Strategy in Computer Graphical Input: An Aid to Visual Design", Ph.D.,  
August 1973.
7. Reed, Walter S.  
"Human Form and Motion Simulation", M.S., January 1970.
8. Stowell, Garrett W.  
"A Two Dimensional Hardware Scheme for Rotation, Translation, and  
Scaling of Computer Graphic Images", M.S.M.E., August 1971.  
  
"Three Dimensional Display Processing Implementation of a Real-Time  
System, Ph.D., December 1974.
9. White, Phillip R.  
"The Interactive Creation and Display of Three Dimensional Graphics",  
Ph.D., May 1975.



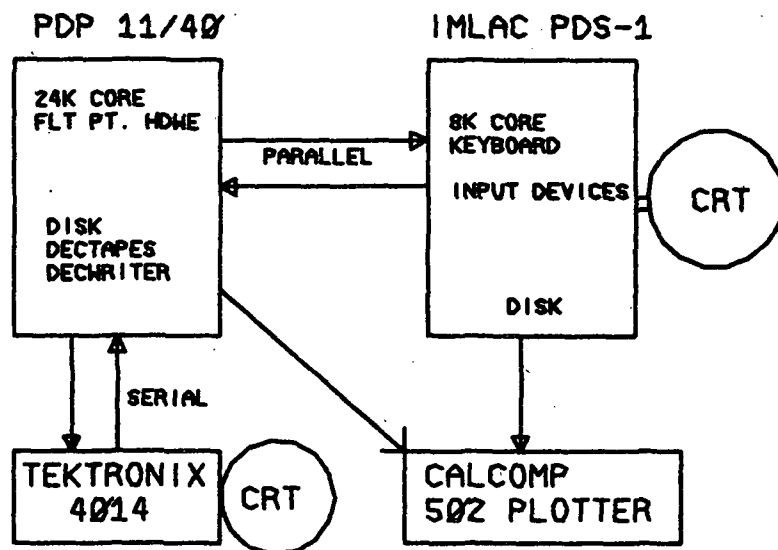


Figure 1. - The equipment configuration.

PURDUE UNIVERSITY

GRAPHICS

Figure 2. - Three-dimensional block letters with hidden lines removed.

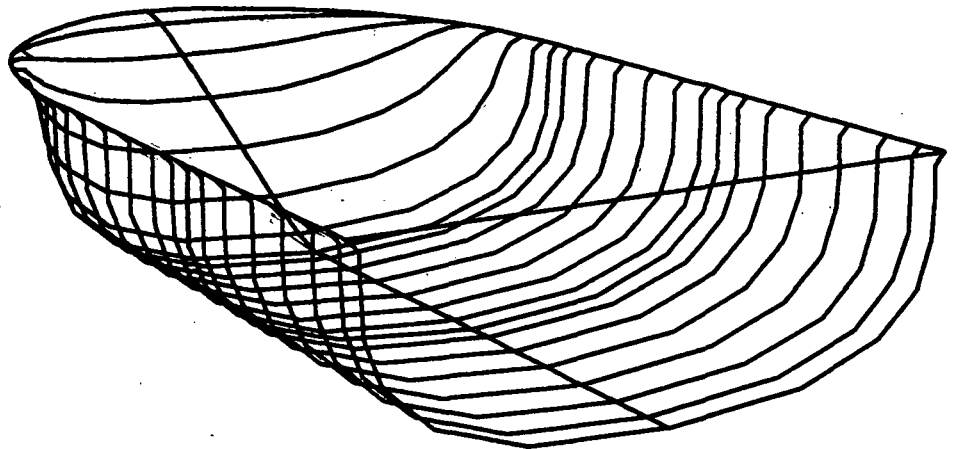
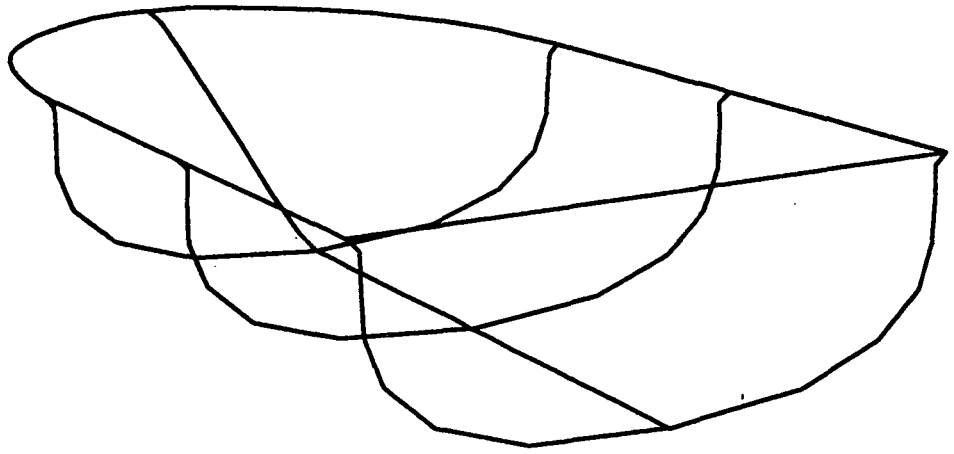


Figure 3. - Two views of a boat hull interactively formed from six surface patches.

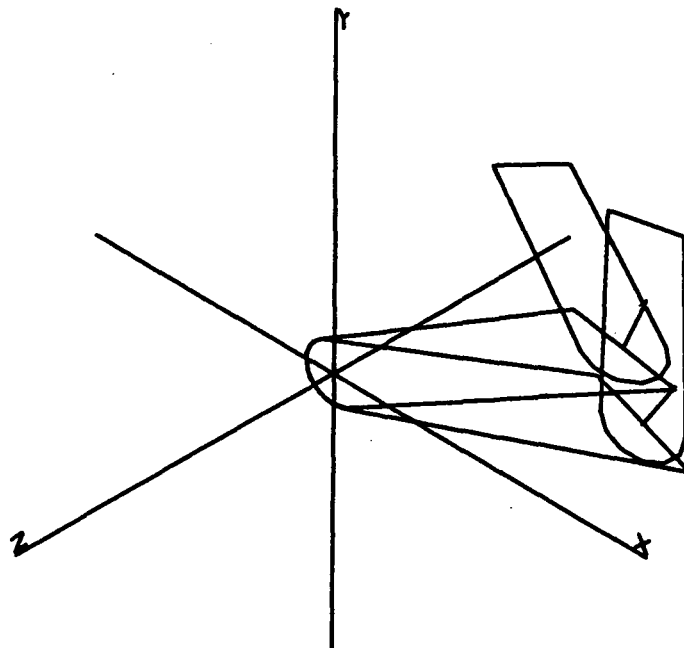


Figure 4. - Two positions of rotation of a three-dimensional linkage.

R/SHIFT R = DECREASE/INCREASE CRANK LENGTH  
L/SHIFT L = DECREASE/INCREASE CONN. ROD LENGTH  
H/SHIFT H = DECREASE/INCREASE OFFSET HEIGHT  
M = MOVE  
F = FASTER  
S = SLOWER  
(HOME) = STOP

R/S = 0.488  
H/S = 0.244  
L/S = 1.220  
TIME RATIO = 1.133  
CAMMAX = 53.13

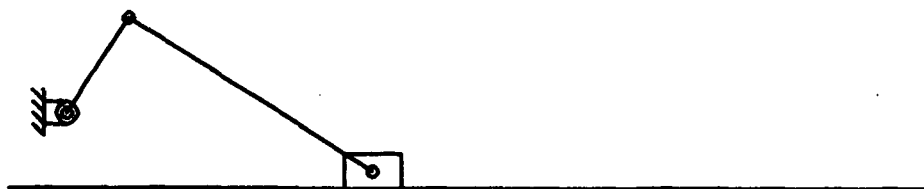


Figure 5. - The interactive motion analysis of an offset slider crank linkage.



## INVERTED SLIDER CRANK STRAIGHT LINE GENERATOR

C = CONTINUOUS MOTION  
 I = INCREMENT WITH ERROR DISPLAY  
 F = FASTER  
 S = SLOWER  
 O = OSCILLATE ON/OFF  
 E = SET ERROR LIMIT  
 B = CHANGE PROPORTIONS  
 (HOME) = STOP

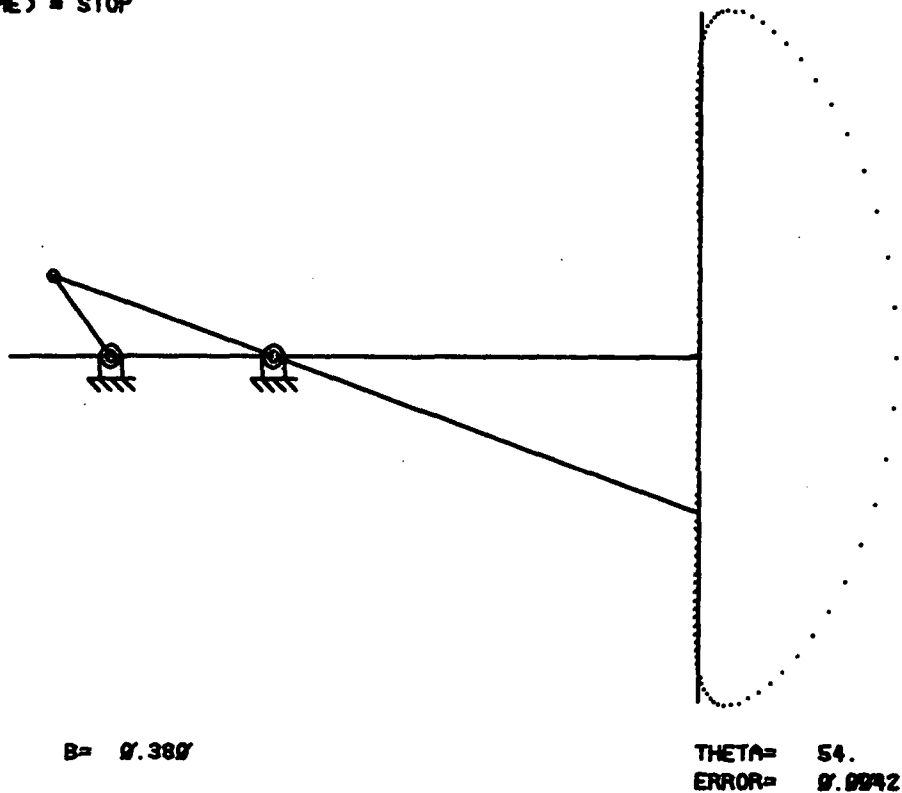


Figure 6. - The interactive dynamic analysis of an inverted slider crank linkage.

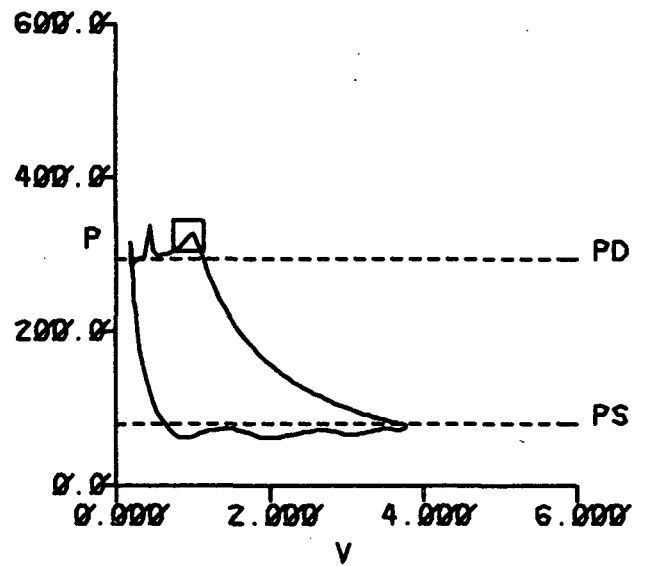
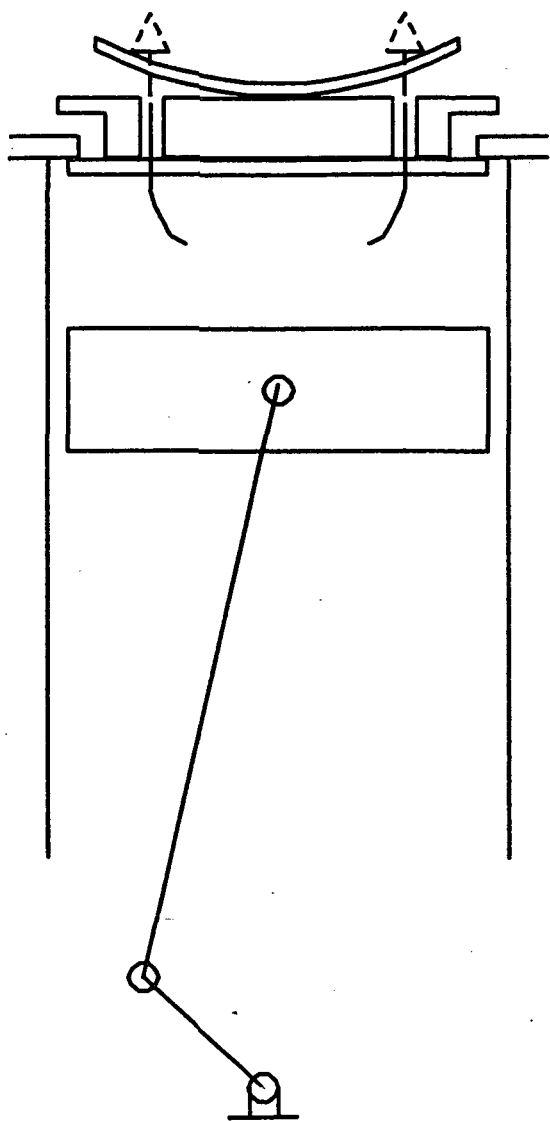
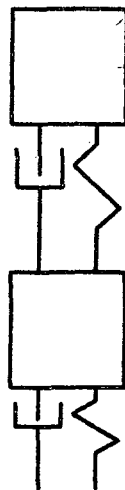
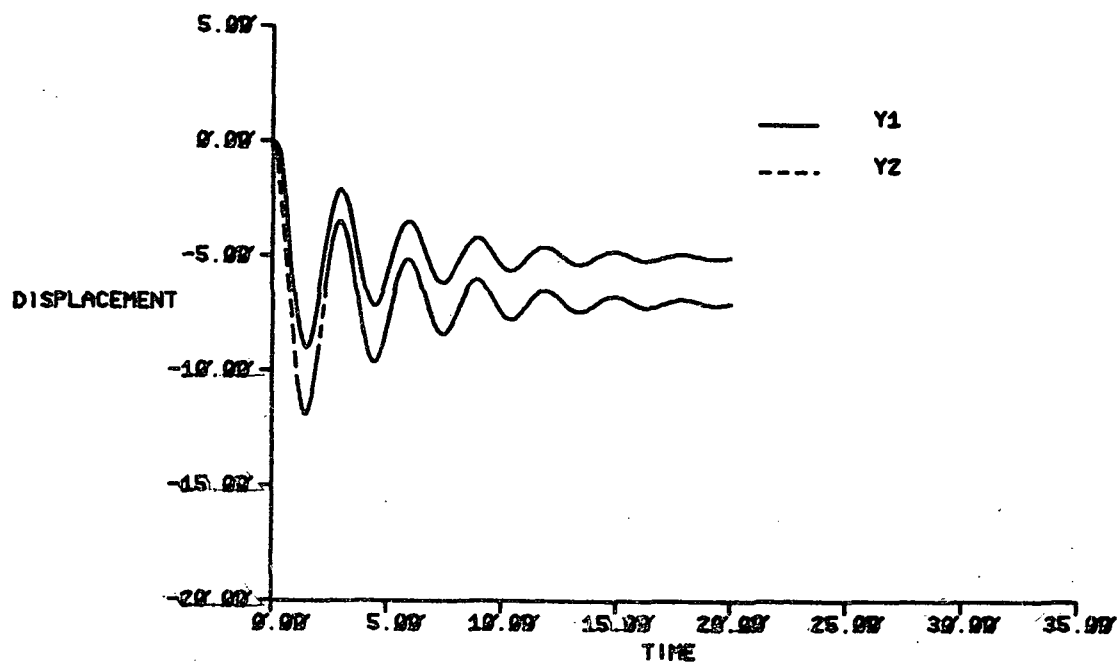


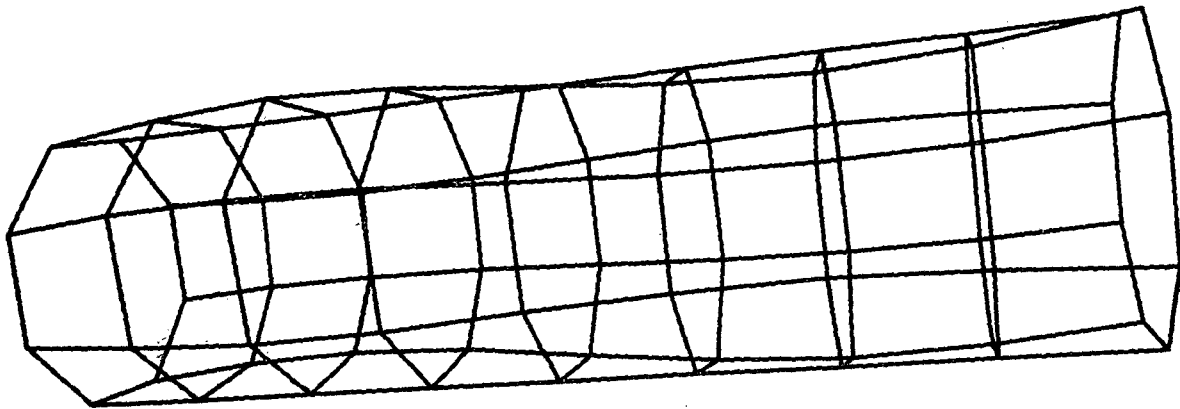
Figure 7. - The graphical display of a computer simulation of a reed valve compressor.



- 1= SEE MOVEMENT
- 2= MODIFY PARAMETERS
- 4= SAVE GRAPH
- 8= GET STORED GRAPH
- 31= EXIT

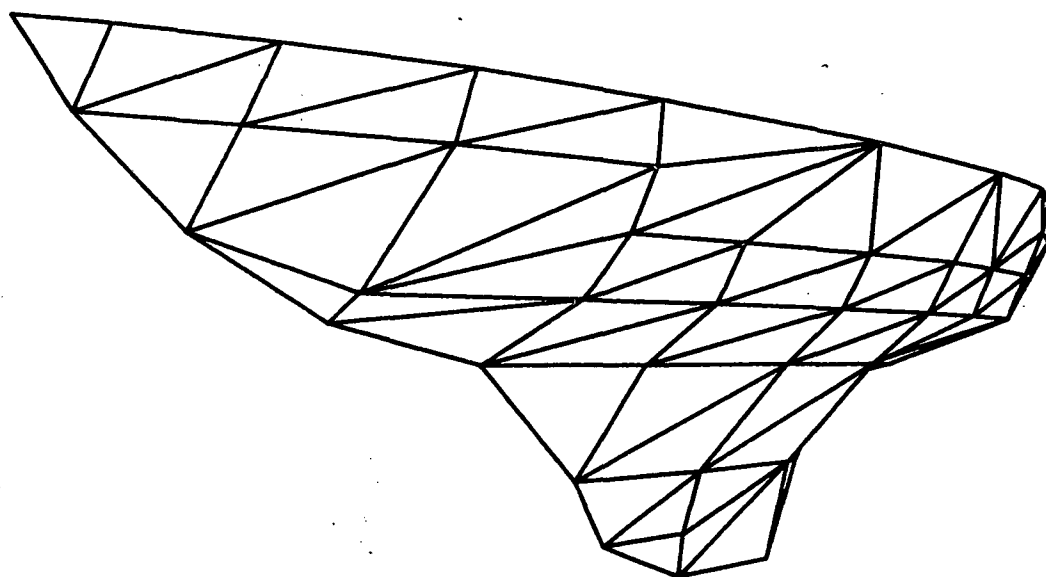
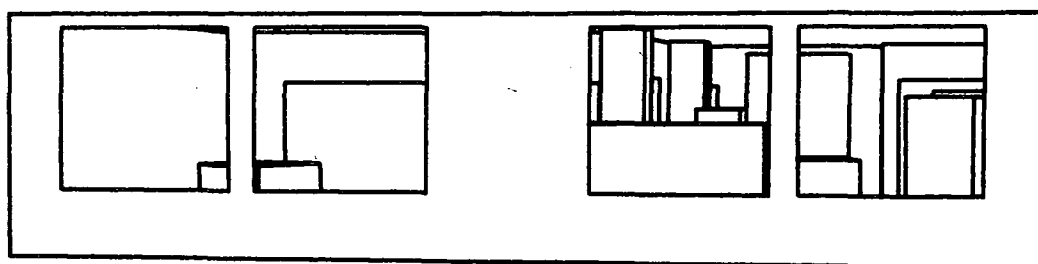
Figure 8. - Interactive display of the vibrations of a spring-mass-damper system.

## KEYSET 3 TO DISPLAY INSTRUCTIONS



M= 2      N= 1      DT= 2.526 MICRO SECONDS

Figure 9. - Interactive three-dimensional display of the free transverse vibration of a cylinder.



1  
;  
~  
Figure 10. - Three-dimensional scenes with hidden lines removed.

# AUTOMATED FINITE ELEMENT GRID BREAK-UP METHOD - A VERIFICATION OF THE SIX NODE AVERAGING APPROACH

H. J. Barten

Florida Research and Development Center

Pratt & Whitney Aircraft

Division of United Technologies Corporation

2

## SUMMARY

The verification of the nodal averaging method for generating automated finite triangular element grids has been demonstrated. This was accomplished with a six node averaging program (SNAP) which was placed on an IBM 2250 vector graphics scope terminal. The advantage of this method is that it is unnecessary to program time consuming geometric division and transition algorithms.

## INTRODUCTION

The most popular method for the automatic break-up of regions into plane finite elements seems to be variations of the "sides-and-parts" method as discussed in references 1 through 4. In this method the structure is divided into four-sided parts with the sides divided into straight line segments connected at nodes. Figures 1, 2, and 3 show such parts in the form of a square which has a one-to-one topological correspondence to a general four-sided figure. If the number of nodes on pairs of opposite sides are equal, lines or curves drawn through corresponding side nodes will generate internal mesh points at their intersections, as shown in figure 1.

For further refinement a division algorithm can be used to specify variable spacing in both directions. If the nodes on one pair of sides are equal in number, while those on the other pair of sides are unequal, this algorithm can be used to generate a transition pattern as shown in figure 2. In this instance the algorithm is that the number of nodes on a row is one less or one more than the number on an adjacent row. In both of these cases the topological pattern is easily predictable and is amenable to automatic mesh generation. If the numbers of nodes on both pairs of sides are not equal, the pattern is generally not too apparent, although exceptions do occur, as shown in figures 3 and 4. The pattern in figure 3 is obtained by drawing the full lines and completing the triangulation by connecting the mesh points at their intersections, as shown by the dashed lines. Here there is a transition from four to five nodes on one pair of opposite sides and a transition from three to four nodes on the other pair. This pattern could have been achieved by using the transition algorithm in two directions and a little imagination. Figure 4 shows the pattern for the same configuration which was obtained by the method which is described herein. This pattern was obtained without specifying



division or transition algorithms and depends only upon the positions of the side nodes. Note that it has the same topological pattern as that shown in figure 3; i. e., the nodes in both figures can be made coincident by shifting without destroying their connectivity.

A more general case in which the division or transition algorithms are more difficult to formulate is shown in figure 5. This pattern was obtained again by the method which is described herein and again required the definition of the position of the side nodes only. Here, the division and transition algorithms are automatic. By inspection it can be seen that the number of nodes along any interior segmented curve is one more or less than the number on adjacent curves, and hence the transition algorithm is satisfied. The transition pattern for this configuration could also have been predicted with some imagination and hind-sight and computerized. However, as the number of side nodes increases, the transition patterns become more difficult to formulate for computer programming.

### NODE AVERAGING APPROACH

This technique was programmed because it makes it unnecessary to use transition and division algorithms or to input internal paths or subsystems. The only rigid condition that is stipulated when using node averaging is that each internal mesh point always be the common vertex of six triangular elements. This is stipulated in order to simplify the computer formulation and to provide uniform nodal patterns at each internal mesh point. If this requirement is met, the topological structure of the break-up is a series of nested polygons each of which has a multiple of six nodes or edges whose number differs by six from the number in adjacent polygons. The reason that this pattern generates the desired transition algorithm is quite obvious if one examines the sector ABCDEFG in figure 6. The number of nodes across the sector increases in increments of 1 as the boundaries of the nested polygons are crossed. Thus, there is one node at A, two nodes along BE, three nodes along CF and finally, four nodes along DG. This algorithm exists in each of the six sectors. The triangulation is achieved by connecting nodes in adjacent polygons as shown in figure 6. Since the external contour of the region is the extreme polygon, the number of known contour nodes or edges also must always be specified as a multiple of six.

We now treat the coordinates of the interior nodes as unknowns and stipulate that they are the average of the coordinates of the six adjacent nodes. This is a rough application of the finite difference method for solving Laplace's equation for a region with prescribed boundary conditions. This algorithm yields two sets of simultaneous equations in which the coordinates of the internal nodes are the unknowns. If the number of nodes is too large for simultaneous equation solutions, the coordinates can be obtained by assuming initial values and iterating until convergence is attained.

The program which has been developed uses the simultaneous equation method. If a hole is present in the region, the number of nodes on the hole contour must also be a multiple of six. The contour of the hole then becomes the first nested polygon while the external contour becomes the last polygon or vice versa. Since the number of nodes on each polygon follows an arithmetic series, the total number of nodes and elements can be expressed in terms of the number

of internal and external contour nodes. If  $N_e$  is the number of external nodes and  $N_i$  is the number of internal nodes, these relations are:

$$\begin{aligned}\text{Total number of nodes} &= \frac{N_e + N_i}{2} + \frac{N_e^2 - N_i^2}{12} \\ \text{Total number of elements} &= \frac{N_e^2 - N_i^2}{6}\end{aligned}$$

If no hole is present, the relations are:

$$\begin{aligned}\text{Total number of nodes} &= 1 + \frac{N_e}{2} + \frac{N_e^2}{12} \\ \text{Total number of elements} &= \frac{N_e^2}{6}\end{aligned}$$

Figures 4 and 5 show the patterns obtained by solving the set of linear simultaneous equations using the node averaging algorithm. Attempts to use this algorithm on regions with concave and/or slender contours were generally unsuccessful because mesh points had a tendency to be near a contour or outside the region. Such regions can be handled by dividing them into subregions and piecing together independent solutions. This technique, however, negates the advantage of a single break-up. Other methods of handling concave boundaries are discussed in the references.

This automatic break-up algorithm has been programmed into a finite element structures program and a compatible finite element break-up scope program. This requires the input of contour nodes only as discussed previously. A test case was input and run on the scope program and a plot of the sequence of presentations is shown in figures 7 through 11. The first display (figure 7) shows the contour nodes generated by inputting corner nodes, hole location, and hole diameter while using an automatic boundary break-up option, which is also incorporated in the program. The second display (figure 8) is the generated automatic break-up. The third display (figure 9) is a refined break-up which is obtained by triangulating the mid-points of the edges of each element in the previous coarse break-up. The fourth display (figure 10) shows the coarse break-up as modified by moving nodes with the light-pen. The fifth display (figure 11) is the fine break-up obtained by triangulating the mid-points of the edges in the previous adjusted break-up. The light-pen can also be used to present magnified local regions, sub-systems, and boundary and perimeter nodes.

## CONCLUDING REMARKS

A nodal pattern for triangular mesh generation, using nodal averaging techniques, has been developed. This pattern has intrinsic properties which make it unnecessary to program internal transition algorithms. An automated mesh generation program using this technique has been developed for use on the

IBM 2250 vector graphics scope terminal. The use of this program on this terminal allows the operator to modify geometry in order to improve the break-up prior to being input into an associated finite element structures program.

#### REFERENCES

1. Buell, W. R., and B. A. Bush: Mesh Generation - A Survey. Transactions of ASME, February 1973, pp. 332-338.
2. Crose, J. G., and R. M. Jones: SAAS II, Finite Element Stress Analysis of Axisymmetric Solids With Orthotropic, Temperature-Dependent Material Properties. TR-0200 (S4980)-1, The Aerospace Corporation, September 1968.
3. Frederick, C. O., Y. C. Wong, and F. W. Edge: Two-Dimensional Automatic Mesh Generation for Structural Analysis. International Journal for Numerical Methods in Engineering, 1970, Volume 2, pp.133-144.
4. Jones, R. E.: QMESH: A Self-Organizing Mesh Generation Program. SLA-73-1088, Sandia Laboratories, July 1974.

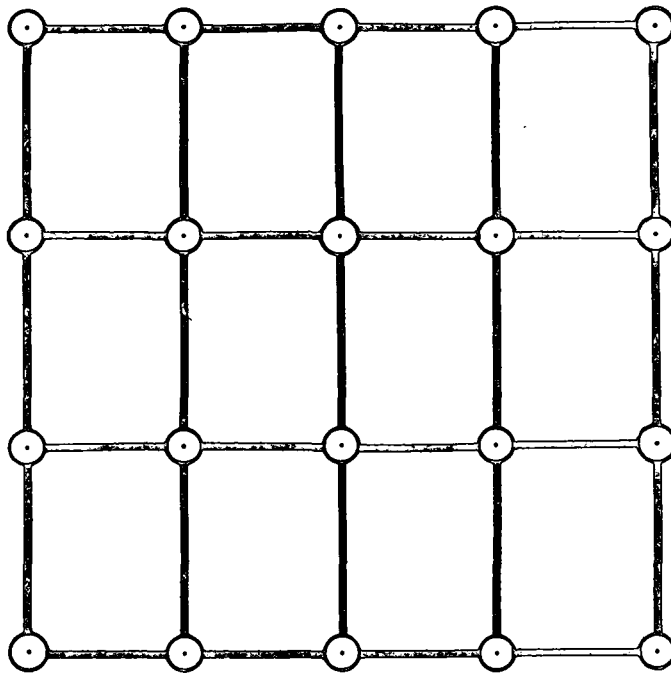


Figure 1. Conventional simple break-up.

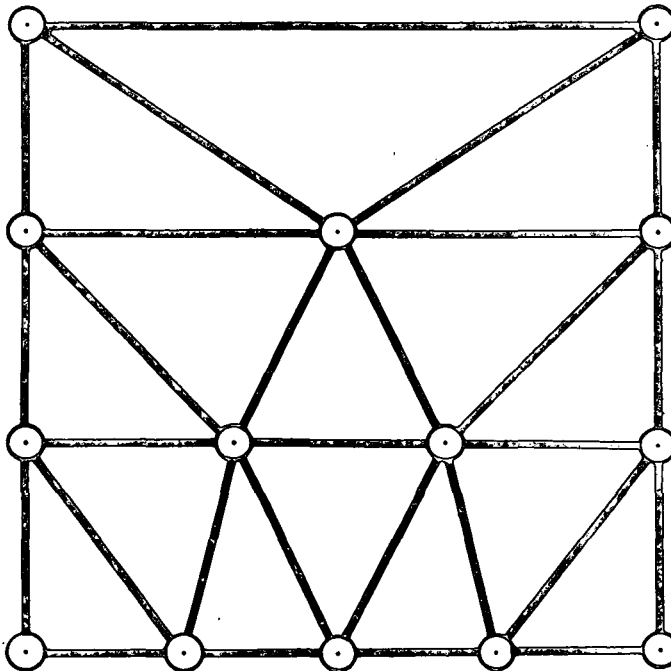


Figure 2. Conventional break-up with single transition.

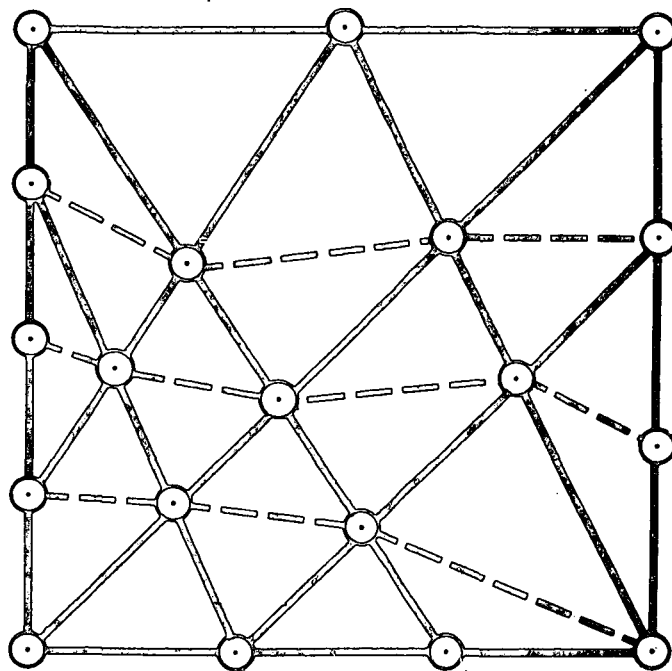


Figure 3. Double transition break-up.

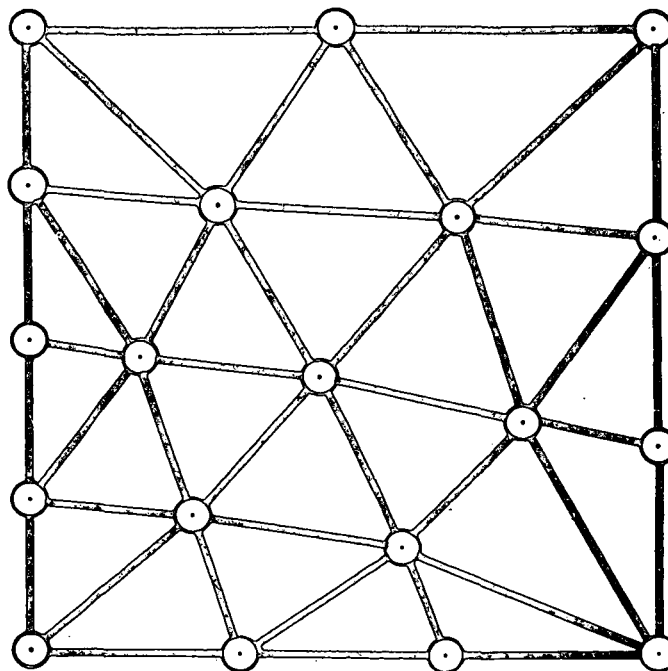


Figure 4. SNAP break-up.

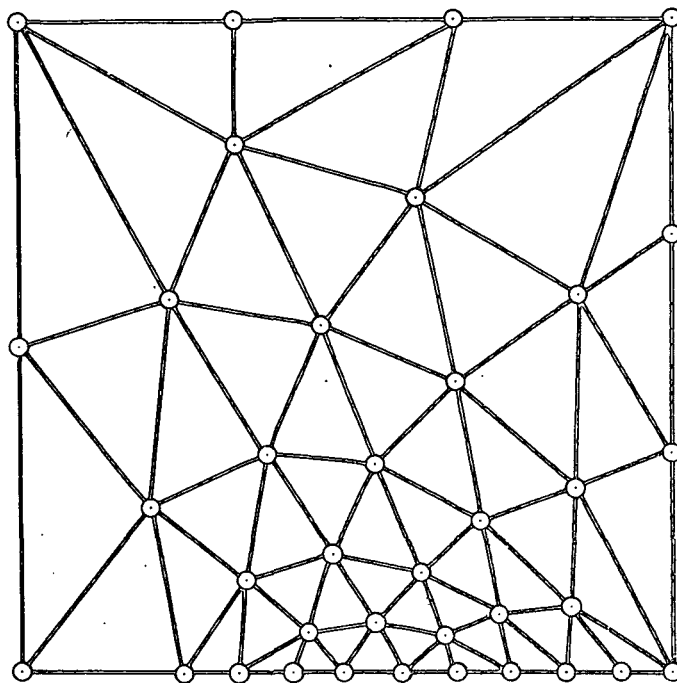


Figure 5. Complex SNAP break-up.

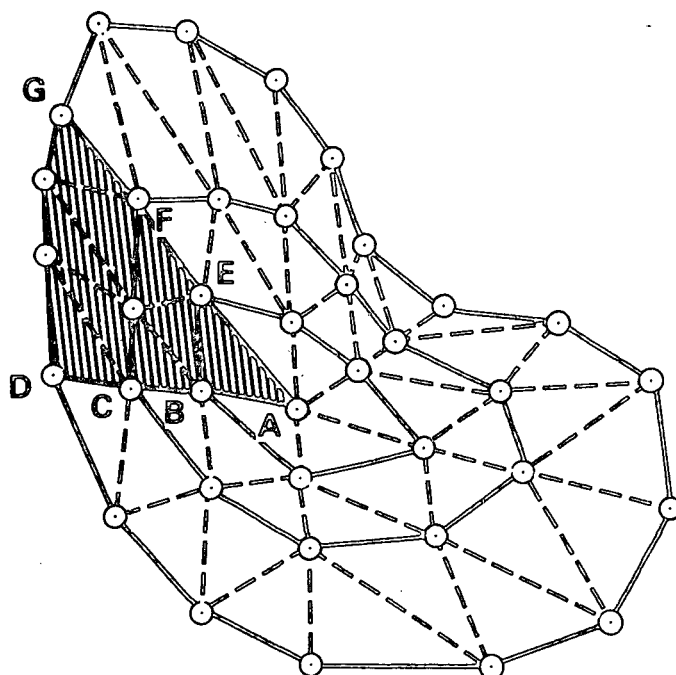


Figure 6. Topology of SNAP break-up.

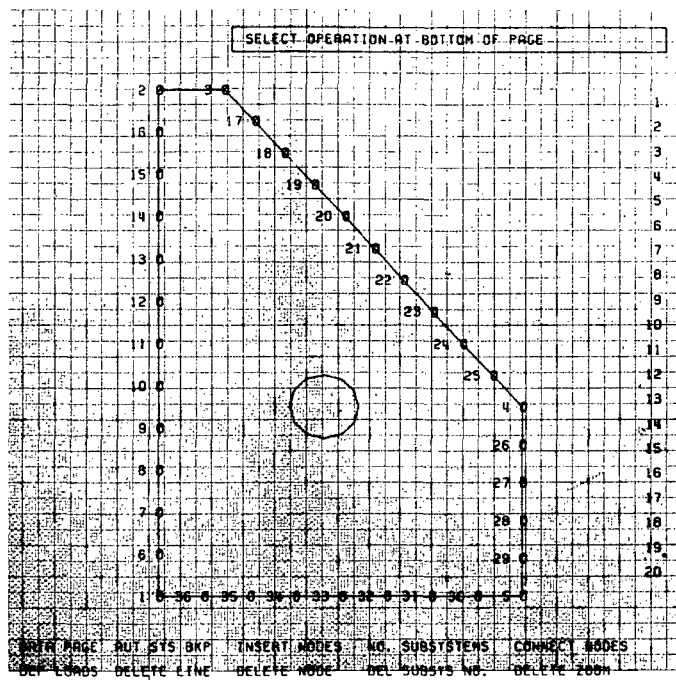


Figure 7. Scope display of boundary.

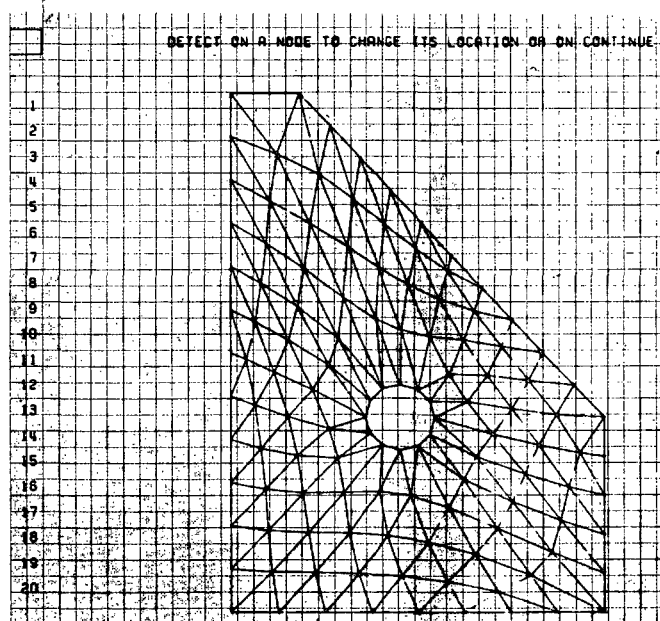


Figure 8. Scope display of coarse SNAP break-up.

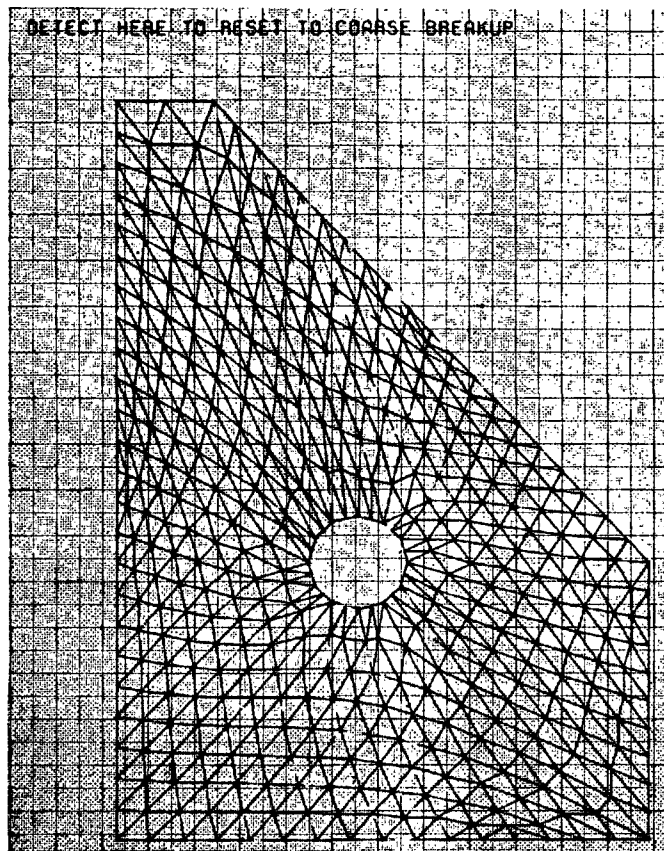


Figure 9. Scope display of fine SNAP break-up.



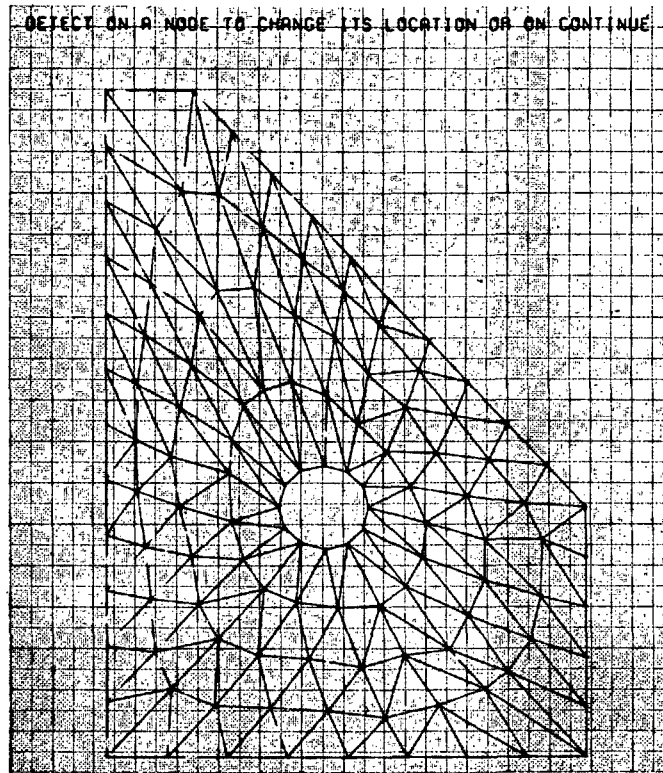


Figure 10. Scope display of adjusted coarse SNAP break-up.

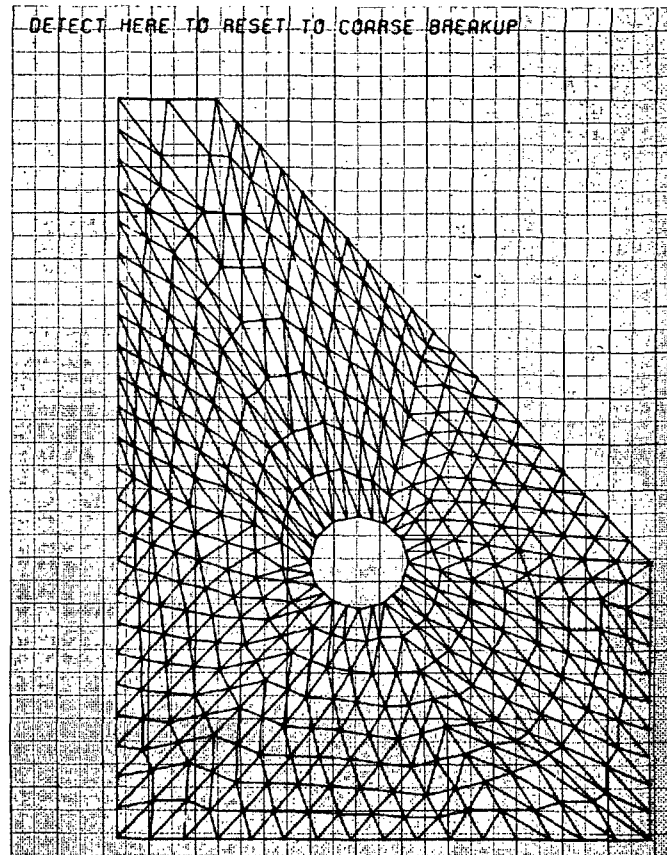


Figure 11. Scope display of adjusted fine SNAP break-up.

**Page Intentionally Left Blank**

# APPLICATION OF THE CINGEN PROGRAM

## A THERMAL NETWORK DATA GENERATOR

William E Schultz and Ronald P. Schmitz

Sperry Support Services

### INTRODUCTION

Since the introduction of the digital computer, analytical simulation of complex physical systems has become a major vocational entity. Although computer speed now makes formerly impossible solutions easily accessible, there is rarely enough lead time between design and fabrication to do a thorough computer analysis. Sperry Support Services engineers have increased their efficiency in analysis of thermal and structural systems by developing computer programs which automatically develop the necessary data for computer simulation and provide visual output to aid the analyst in verification of the model. The primary program for discussion in this paper is entitled CINGEN (Ref. 1), a contraction of CINDA data GENEerator, after the CINDA thermal analysis program (Ref. 2). Supporting programs, COON3D (Ref. 3) and GEOMPLT (Ref. 4) developed for structural analysis and adapted for use with CINGEN, will also be discussed.

### CINGEN TECHNIQUE

Present state-of-the-art mathematical modeling techniques for evaluation of structural and thermal performance of physical systems are distinctly different. The leading method for structural analysis is the finite element approach. The finite differencing technique is most frequently used for thermal analysis. Although thermal problems can be solved by finite element analysis (See Refs. 5 and 6), present state-of-the-art finite element programs cannot compete in computer execution speed with conventional thermal programs using the finite differencing method. Frequently separate thermal and structural math models must be developed which have a one-to-one positional correspondence to allow thermal loads to be evaluated in the structural analysis (Ref. 7). The duplication of modeling effort can be quite costly, especially if large systems are involved. Avoidance of this duplication effort and a desire for a geometrical representation of the thermal model to reduce modeling errors were key factors which prompted the development of CINGEN.

The CINGEN program simplifies the thermal modeling process by performing all of the capacitance and conductance calculations normally done by the analyst. Each solid element is divided into five tetrahedrons, allowing the total volume to be calculated precisely. The thermal capacitance is then calculated as the product of volume, density and specific heat. The center of gravity of each element is calculated, and the thermal resistance is based on the distance between element centroids divided by the product of the element interface area

and the thermal conductivity. There is no problem with elements of dissimilar materials having a common interface, since the total conductance between the elements is the reciprocal of the sum of the thermal resistances for each node to the interface.

Mathematical modeling of thermal systems is somewhat of an art and reliability depends greatly on the principle of subdivision of the system. A thermal analyst can use CINGEN without prior experience with finite element modeling techniques and without the aid of supporting programs. He needs only to determine the physical location of desired coordinates for each desired element. The sample problem chosen includes both manually developed and automatically developed input for CINGEN.

### SAMPLE PROBLEM

Figure 1 lists seven basic steps in the utilization of CINGEN, and supporting programs COON3D and GEOMPLT. COON3D was developed by Sperry Support Services to automatically generate a three-dimensional solid-element grid mesh of desired systems. The program has the ability to generate multilayer models as exhibited in the sample problem. Figure 2 is the COON3D input which describes the four bounding curves for each surface. Figure 3 contains the COON3D output and the manually developed input for CINGEN. For verification of the COON3D model, program GEOMPLT is used to display the model on the CRT graphics screen. Figure 4 contains the GEOMPLT user options as viewed on the CRT screen. Figure 5 displays the GEOMPLT user option of plotting NASTRAN bulk data which was generated by COON3D and by manual development, and the option to plot the thermal network created by CINGEN. Figure 6 contains a sketch of the actual hardware and a side view of the CINGEN model.

When the analyst is satisfied with the appearance of the COON3D generated model, CINGEN is executed. Figure 7 contains the CINGEN output which is a format acceptable to the thermal analyzers CINDA, SINDA, and MITAS (Refs. 2, 8, and 9). Data files are also created which allow the analyst to view the thermal network created by CINGEN. Figures 8(a) to 8(h) are a series of partial views of the thermal network displayed by GEOMPLT. If the analyst is satisfied with the model, he can use the Text Editor processor capability of UNIVAC 1100 series computers to input the system boundary conditions and then execute SINDA. The resulting SINDA solution can be viewed using the special purpose line printer plotter routine (Ref. 10) for X-Y plotting, Figure 9.

### CONCLUDING REMARKS

With the aid of these analytical tools - CINGEN, COON3D, GEOMPLT - analysis cost and time can be drastically reduced from manual development of models. In addition, the visual verification of the models provides added confidence in the accuracy of the model. These tools will lead to reduced time between design, analysis, and fabrication and contribute to final development of the best design rather than one which is expedient.

## REFERENCES

1. Schultz, W. E.: "Automated Data Generation for Thermal/Structural Models," Attachment 1. Sperry Support Services Memorandum, Huntsville, Alabama, October 17, 1974.
2. Gaski, J. D., Lewis, D. R., and Thompson, L. R.: "Chrysler Improved Numerical Differencing Analyzer for Third Generation Computers," TN-AP-67-287, October 20, 1967.
3. Chan, G. C.: "Automated Data Generation for Thermal/Structural Models," Attachment 2. Sperry Support Services Memorandum, Huntsville, Alabama, October 17, 1974.
4. Schmitz, Ronald P.: "GEOMPLT - Interactive Graphics Program for Finite Element and Thermal Network Models." Sperry Support Services, Huntsville, Alabama, November 30, 1974.
5. Zienkiewicz, O. C.: The Finite Element Method in Engineering Science, Second Edition, McGraw-Hill Pub., Ltd., Berkshire, England, 1971.
6. McCormick, C. W.: Editor. "The NASTRAN User's Manual," (Level 15.6). NASA SP-222(01), January 1975. (Revised by Sperry Support Services, Huntsville, Alabama.)
7. Loafman, J. W., Schmitz, R. P., and Eldrige, C. M.: "NASTRAN Thermo-structural Analysis of a High Energy Laser Mirror and Comparison with CINDA Thermal Analysis," Fifth Annual Navy-NASTRAN Colloquium." September 10, 1974.
8. Smith, J. P.: "SINDA User's Manual." NASA Contract 9-10435, TRW System Group, April 1971.
9. "Martin Interactive Thermal Analysis System." MDS-SPLPD-71-FD238, Martin Marietta Corporation Denver Data Center, Denver, Colorado, March 1971.
10. Schultz, W. E., and Stephen, L. A.: "A Self-Contained Line Printer Plotting Routine." Sperry Support Services Memorandum, Huntsville, Alabama, May 30, 1974.

STEPS	FUNCTION	HARDWARE	SOFTWARE
1	GENERATE FINITE ELEMENT MODEL	TTY	COON3D/MANUAL
2	DISPLAY MODEL	CRT	GEOMPLT/PLOT MODE
3	EDIT MODEL	CRT	GEOMPLT/EDIT MODE
4	GENERATE THERMAL NETWORK	TTY	CINGEN
5	DISPLAY THERMAL NETWORK	CRT	GEOMPLT/THERMAL NETWORK PLOT
6	SINDA DECK SET UP	TTY	FUR/PUR
7	SINDA ANALYSIS	TTY (BRKPT)	SINDA

Figure 1.- Basic steps in the use of CINGEN.

SPACE SHUTTLE HYDRAULIC ENGINE ACTUATOR INTERFACE RING

CURVE U 0

CURVE U 1

X

Y

Z

X

Y

Z

1	0.00000	2.30000+00	-6.50000+00	2.00000+00	0.00000	-6.50000+00
2	0.00000	3.30000+00	-6.50000+00	3.30000+00	0.00000	-6.50000+00

CURVE W 0

CURVE W 1

X

Y

Z

X

Y

Z

1	0.00000	2.00000+00	-6.50000+00	0.00000	3.30000+00	-6.50000+00
2	1.40000+00	1.40000+00	-6.50000+00	3.30000+00	2.30000+00	-6.50000+00
3	2.00000+00	0.00000	-6.50000+00	3.30000+00	0.00000	-6.50000+00

Figure 2.- Sample COON3D input data.



C1530A	3004	5204	5203	5205	5206	5304	5303	5305	3004	Manual
3004	5304									
C1530A	3005	5301	5302	5303	5304	5401	5402	5403	3005	
3005	5404									
C1530A	3006	5304	5303	5305	5306	5404	5403	5405	3006	
3006	5406									
C1530B	3007	5401	5402	5403	5404	5501	5502	5503	3007	Manual
3007	5504									
C1530B	3008	5404	5403	5405	5406	5504	5503	5505	3008	
3008	5506									
C1530B	10001	10001	10002	10005	10004	10101	10102	10105	10001	
10001	10104									
C1530B	10002	10002	10003	10006	10005	10102	10103	10106	10002	COON3D
10002	10105									
C1530B	10003	10004	10005	10008	10007	10104	10105	10108	10003	
10003	10107									
C1530B	10004	10005	10006	10009	10008	10105	10106	10109	10004	
10004	10108									
C1530B	10101	10101	10102	10105	10104	10201	10202	10205	10101	Manual
10101	10204									
GRID	5501		-.420	-.420	3.530					
GRID	5502		-.420	-.420	3.530					
GRID	5503		-.420	-.000	3.530					
GRID	5504		-.420	-.000	3.530					
GRID	5505		-.420	-.420	3.530					COON3D
GRID	5506		-.420	-.420	3.530					
GRID	10001		.000	2.000	-6.500					
GRID	10002		1.400	1.400	-6.500					
GRID	10003		2.000	.000	-6.500					
GRID	10004		.000	2.000	-6.500					
GRID	10005		1.850	1.850	-6.500					Manual
GRID	10006		2.450	.000	-6.500					

Figure 3.- Sample element and grid data automatically developed by COON3D and manually developed input for CINGEN.

EXQT GEOM MPA

```

  X X
  X X
 XX XX
XXXXX XXXXX
XXXXX XXXXX
  XX XX
  X X
  X X

```

```

XXXXXXXXXXXXXXXX XXXXX XXXXXXXXXXXXXXXX X X
X X X X X X X X X X
XXXX XXXXX XXX XXXXX XXX XXX
  X X X X X X X
XXXX X XXXXX X XX XX X

```

INTERACTIVE GRAPHICS PROGRAM  
 \*\*GEOMPLT\*\* VERSION 3.0  
 GENERATED BY SPERRY SUPPORT SERVICES  
 NOV 74

```

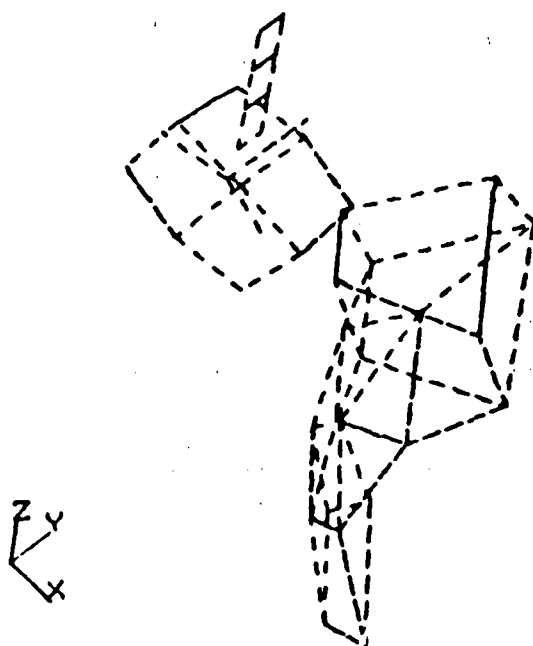
DIFFERTIAL TRANSDUCER MANUALLY DEVELOPED
** TYPE ( $INPUT (OPTIONS) $)
** NOTE- 1ST COLUMN IS IGNORED FOR NAMELIST INPUT.
**      (.) AS LAST CHARACTER TO CONTINUE ON NEXT LINE,
**      ($) AS LAST CHARACTER TERMINATES NAMELIST INPUT
INPUT PROGRAM=3.NODENO=1.PU=1$
** TYPE ( $PVIEW JPU(1)= .(ELEMENT NUMBERS) $)
** TO DEFINE PARTIAL VIEW

** NOTE - 1ST COLUMN IS IGNORED FOR NAMELIST INPUT.
**      (.) AS LAST CHARACTER TO CONTINUE ON NEXT LINE,
**      ($) AS LAST CHARACTER TERMINATES NAMELIST INPUT
**      (-0 OR 4HTHRU) IS USED FOR CONSECUTIVE NUMBERS
**      EXAMPLE 1,-0,4= 1.2.3,4.

```

Figure 4.- GEOMPLT header page and typical user options.

TOTAL RESISTANCE NETWORK  
\*\* TYPE (C) TO CONTINUE



TOTAL RESISTANCE NETWORK  
\*\* TYPE (C) TO CONTINUE

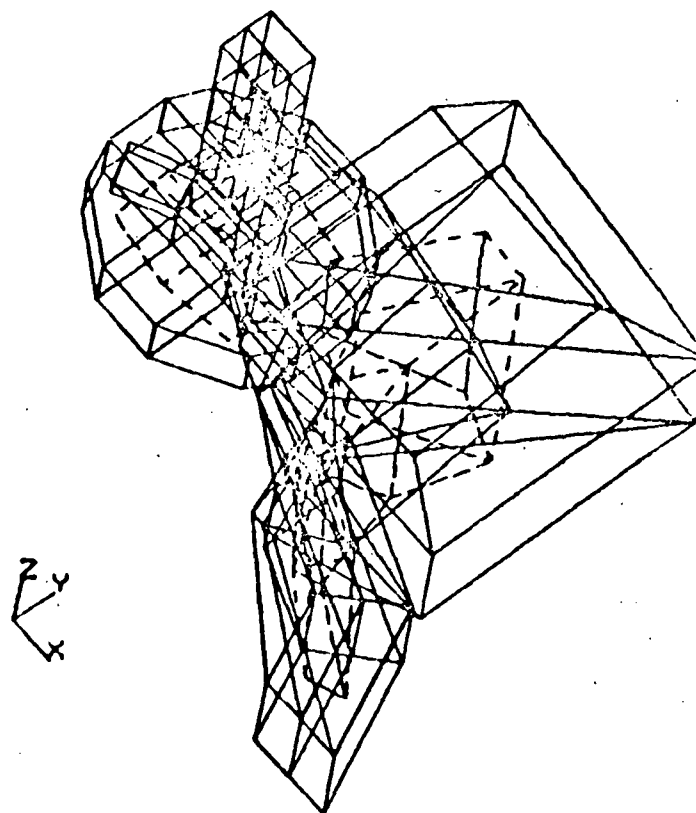


Figure 5.- Total model NASTRAN type bulk data and resistance network.

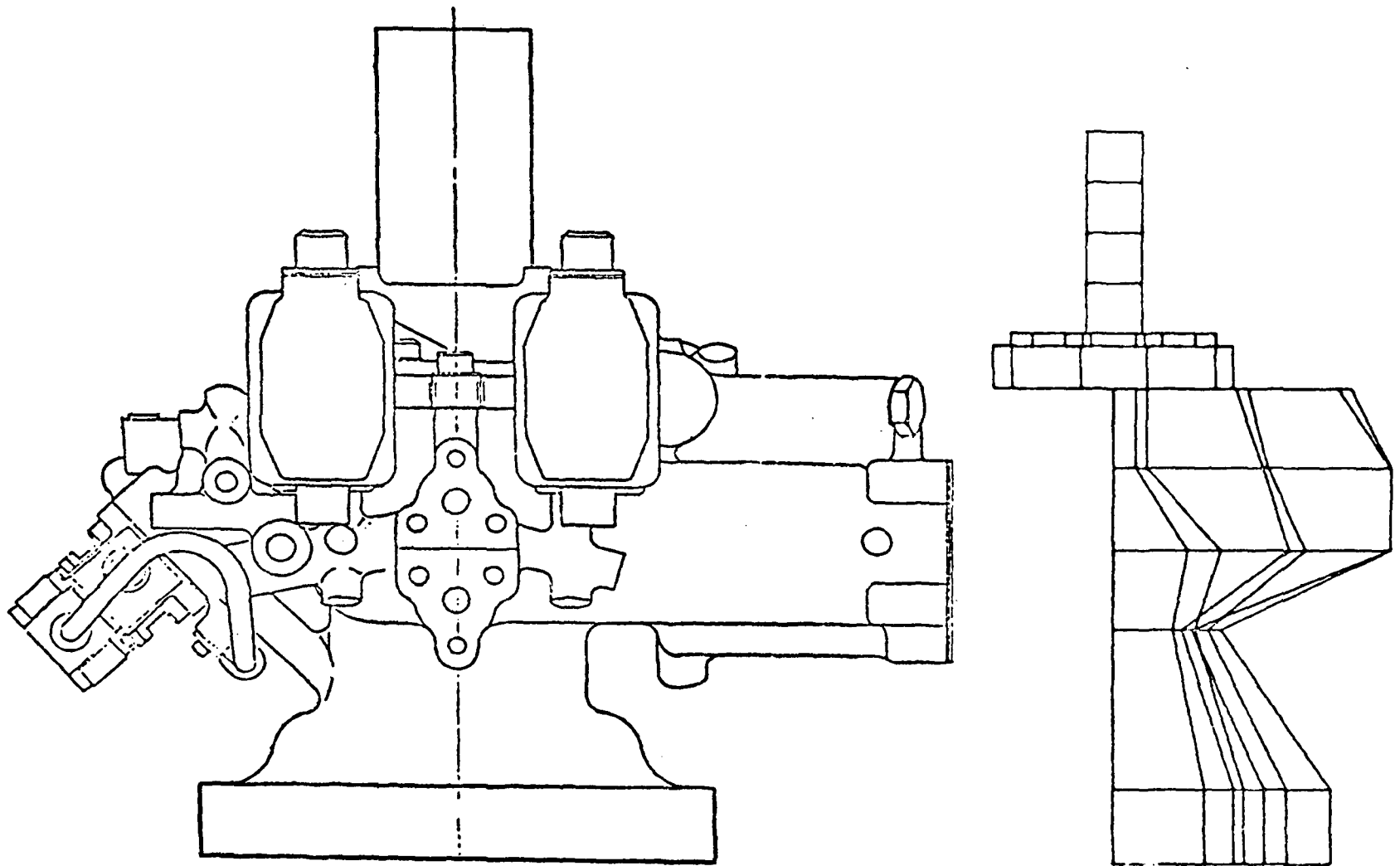


Figure 6.- Sketch of the actuator and the CINGEN model.

CINDA MODEL BEGINS WITH A BLANK CARD

BCD 3THERMAL SPCS		
BCD 6TITLE= SPACE SHUTTLE HYDRAULIC ENGI		
BCD 6NE ACTUATOR INTERFACE RING		
END		
BCD 3NODE DATA		
1001,100.,		.01262
1002,100.,		.01666
1003,100.,		.01666
1004,100.,		.01666
1005,100.,		.01666
1006,100.,		.01383
1007,100.,		.01383
1008,100.,		.01383
1009,100.,		.01383
2001,100.,		.00287
2002,100.,		.00277
2003,100.,		.00230
2004,100.,		.00230
2005,100.,		.00227
3001,100.,		.00557
3002,100.,		.00557
3003,100.,		.00557
3004,100.,		.00557
3005,100.,		.00557
3006,100.,		.00557
3007,100.,		.00557
3008,100.,		.00557
10001,100.,		.02370
10002,100.,		.02370
10003,100.,		.03029
10004,100.,		.03029
10101,100.,		.02796
10102,100.,		.02919
10103,100.,		.03436
10104,100.,		.03655
10201,100.,		.04216
10202,100.,		.03919
10203,100.,		.06521
10204,100.,		.06513
10301,100.,		.06831
10302,100.,		.07844
10303,100.,		.17928
10304,100.,		.15652
10401,100.,		.06087
10402,100.,		.05647
10403,100.,		.14438
10404,100.,		.13388
END		

Figure 7.- Computer printout of the model generated by CINGEN.

# BCD 3CONDUCTOR DATA

1,	1001,	1002,	3.86207
2,	1001,	1003,	3.86207
3,	1001,	1004,	3.86207
4,	1001,	1005,	3.86207
5,	1002,	1006,	5.51663
6,	1002,	1009,	5.51663
7,	1003,	1007,	5.51663
8,	1003,	1008,	5.51663
9,	1004,	1008,	4.15913
10,	1004,	1009,	4.15913
11,	1005,	1006,	4.15913
12,	1005,	1007,	4.15913
13,	2001,	2002,	2.40120
14,	2001,	2003,	.82993
15,	2001,	2004,	.82993
16,	2002,	2005,	1.43174
17,	3001,	3002,	11.20000
18,	3001,	3003,	4.03200
19,	3002,	3004,	4.03200
20,	3003,	3004,	11.20000
21,	3003,	3005,	4.03200
22,	3004,	3006,	4.03200
23,	3005,	3006,	11.20000
24,	3005,	3007,	4.03200
25,	3006,	3008,	4.03200
26,	3007,	3008,	11.20000
27,	10001,	10002,	2.68636
28,	10001,	10003,	26.72456
29,	10001,	10101,	2.92444
30,	10002,	10004,	26.72456
31,	10002,	10102,	3.03076
32,	10003,	10004,	2.10140
33,	10003,	10103,	3.20498
34,	10004,	10104,	3.37718
35,	10101,	10102,	5.12264
36,	10101,	10103,	71.09416
37,	10101,	10201,	2.66558
38,	10102,	10104,	74.85917
39,	10102,	10202,	2.80583
40,	10103,	10104,	4.19219
41,	10103,	10203,	2.04396
42,	10104,	10204,	2.39383
43,	10201,	10202,	6.07447
44,	10201,	10203,	15.65426
45,	10201,	10301,	8.18221
46,	10202,	10204,	20.24132
47,	10202,	10302,	8.40394
48,	10203,	10204,	3.93730
49,	10203,	10303,	4.27694
50,	10204,	10304,	5.56573
51,	10301,	10302,	9.93401
52,	10301,	10303,	8.42990
53,	10301,	10401,	16.28932
54,	10302,	10304,	9.99392

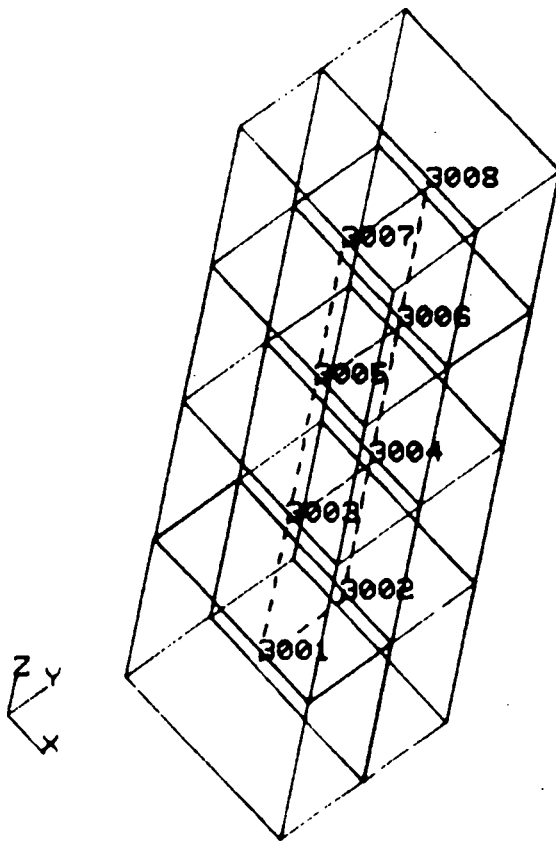
Figure 7.- Continued.

	55,10302,10402,	15.41091
	56,10303,10304,	4.60041
	57,10303,10403,	27.92661
	58,10304,10404,	33.33628
	59,10401,10402,	13.20353
	60,10401,10403,	6.82662
	61,10402,10404,	7.37689
	62,10403,10404,	5.27584
END		
BCD 3	CONSTANTS DATA	
END		
	TIMEND,0.01,OUTPUT,0.001,ITEST,0	
BCD 3	ARRAY DATA	
	-100,SPACE,100,END \$ TIME ARRAY	
	-101,SPACE,100,END \$ TEMP ARRAY	
	-102,SPACE,100,END \$ TEMP ARRAY	
200		
BCD 6	TIME IN HOURS	
BCD 6		,END
201		
BCD 6	TEMPERATURE -DEGREES F	
BCD 6		,END
202		
BCD 6	LINE PRINTER PLOT OF SAMPLE PROBLEM	
BCD 6		,END
END		
BCD 3	EXECUTION	
F	DIMENSION X(4000)	
F	NDIM=4000	
F	NTH=0	
	CNFRWD	
	PLOTIT(0,2,40,70,ITEST,A100,A101,A202,A200,A201)	
	PLOTIT(1,2,40,70,ITEST,A100,A102,A202,A200,A201)	
END		
BCD 3	VARIABLES 1	
END		
BCD 3	VARIABLES 2	
END		
BCD 3	OUTPUT CALLS	
	TPRINT	
F	ITEST=ITEST+1	
	STOARY(ITEST,TIMEN,A100) \$ TIME	
	STOARY(ITEST,T10001,A101) \$ RING	
	STOARY(ITEST,T10101,A102) \$ CONE	
END		
BCD 3	END OF DATA	

Figure 7.- Concluded.

RUDT DIFFERENTIAL TRANSDUCER MANUALLY DEVELOPED

\*\* TYPE (C) TO CONTINUE



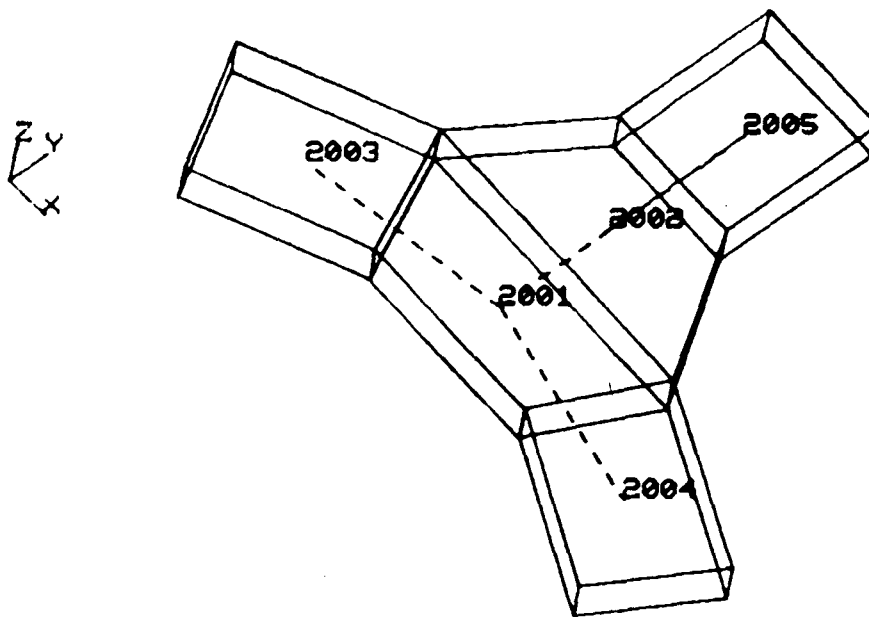
(a) Partial view of the RVDT.

Figure 8.- Resistance network.



MOUNTING BRACKET MANUALLY DEVELOPED

\*\* TYPE (C) TO CONTINUE

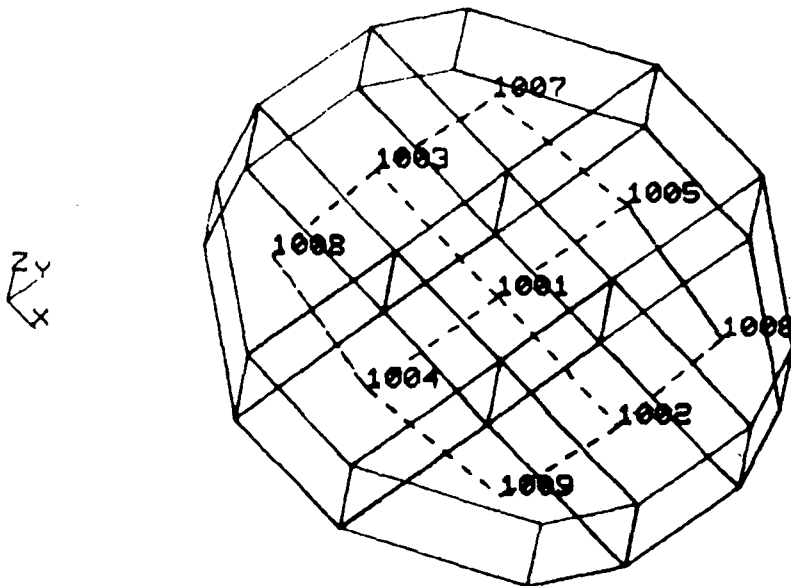


(b) Partial view of the mounting bracket.

Figure 8.- Continued.

TOP PLATE 360 DEGREE MANUALLY DEVELOP  
ED GRID

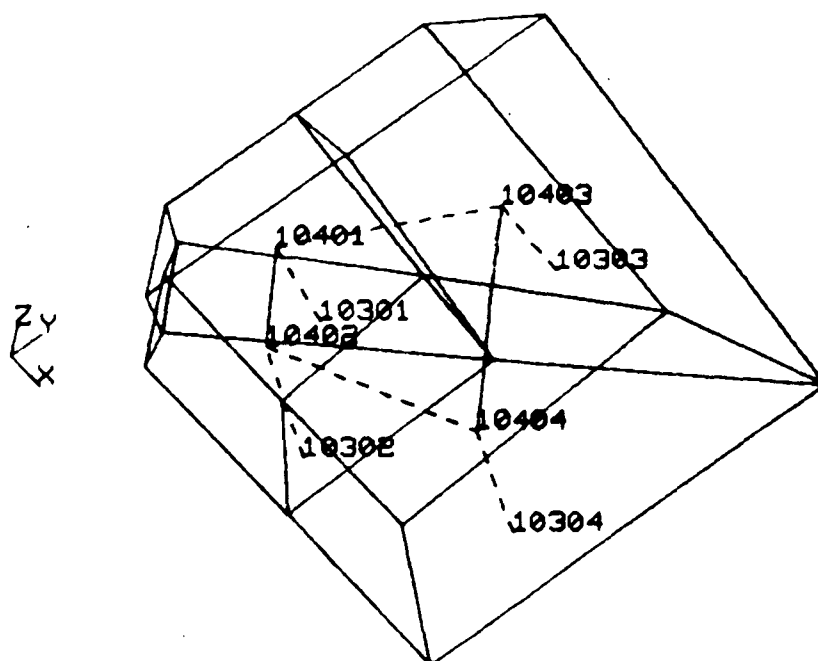
\*\* TYPE (C) TO CONTINUE



(c) Partial view of the top plate.

Figure 8.- Continued.

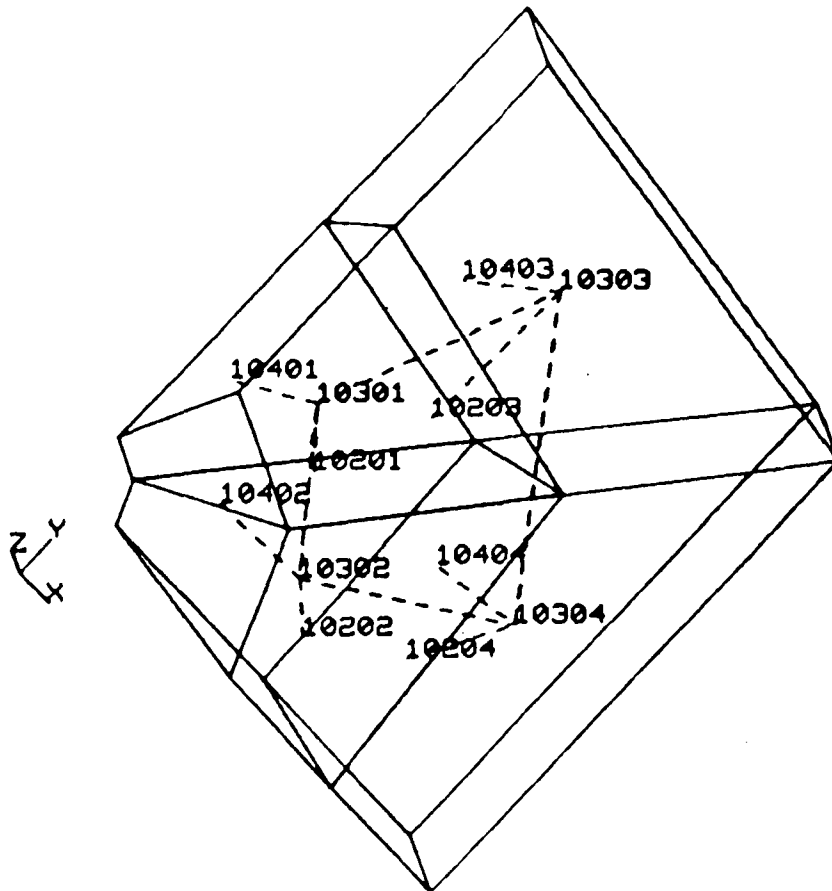
BODY SEGMENT NUMBER 3  
\*\* TYPE (C) TO CONTINUE



(d) Partial view of body segment 3.

Figure 8.- Continued.

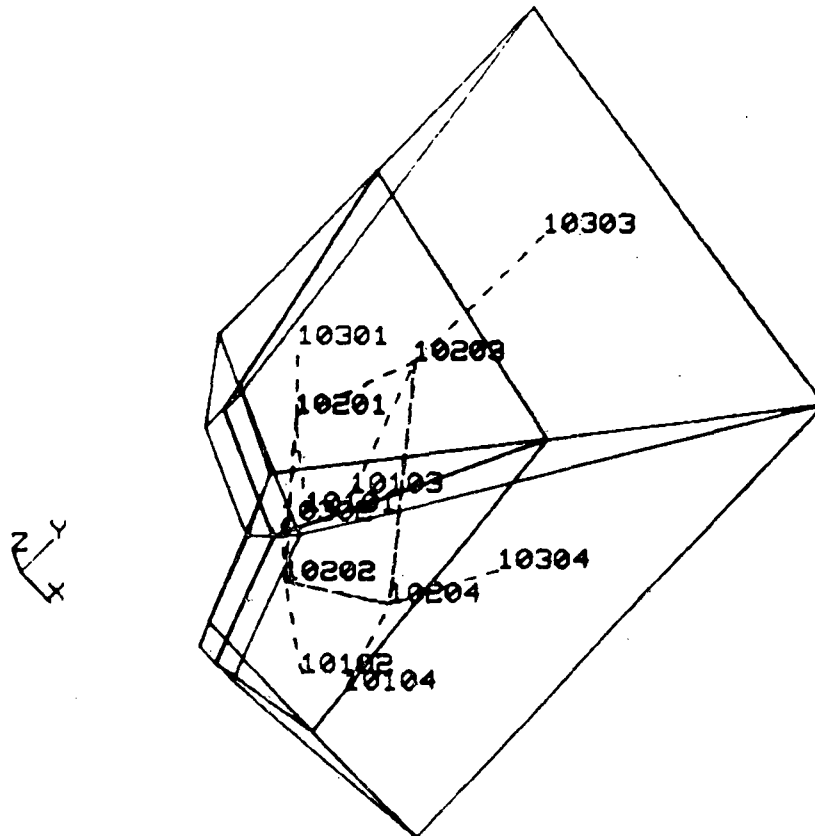
BODY SEGMENT NUMBER 2  
 \*\* TYPE (C) TO CONTINUE



(e) Partial view of body segment 2.

Figure 8.- Continued.

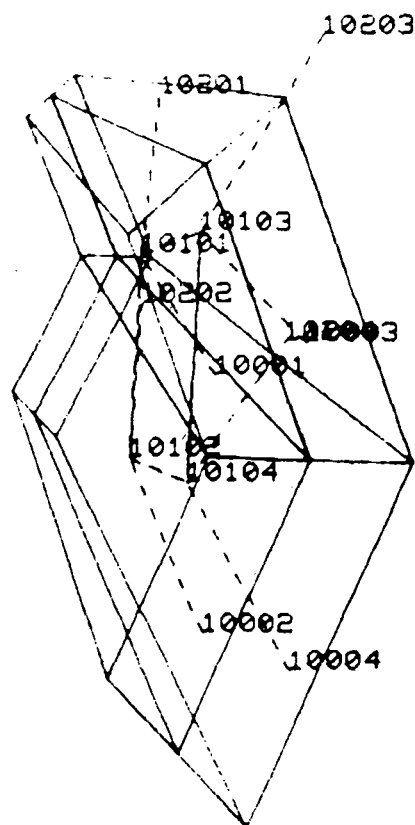
BODY SEGMENT NUMBER 1  
\*\* TYPE (C) TO CONTINUE



(f) Partial view of body segment 1.

Figure 8.- Continued.

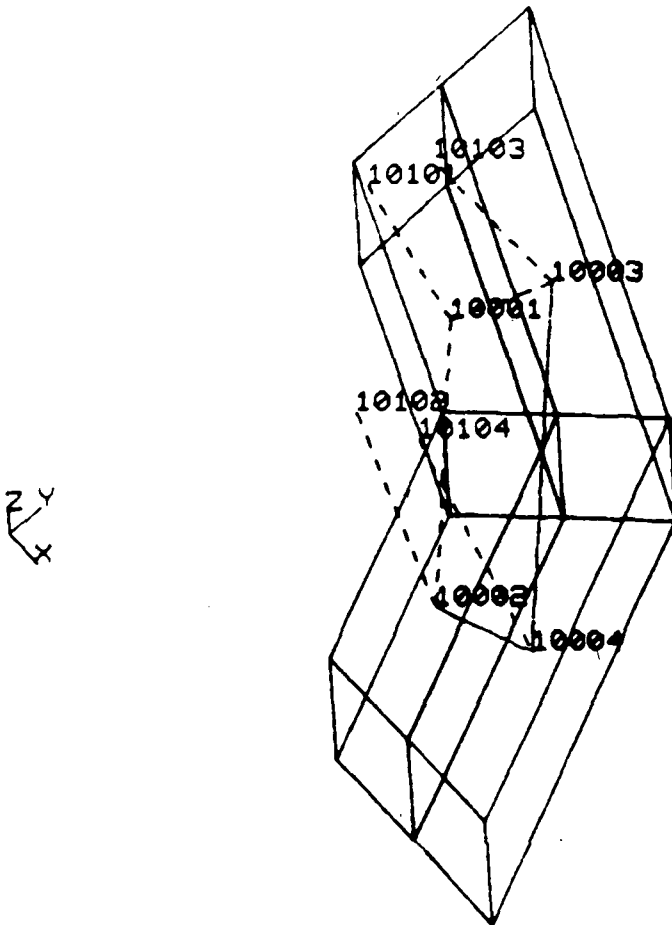
PISTON CONE  
 " TYPE (C) TO CONTINUE



(g) Partial view of the piston cone.

Figure 8.- Continued.

INTERFACE RING RESISTANCE NETWORK  
 \*\* TYPE (C) TO CONTINUE



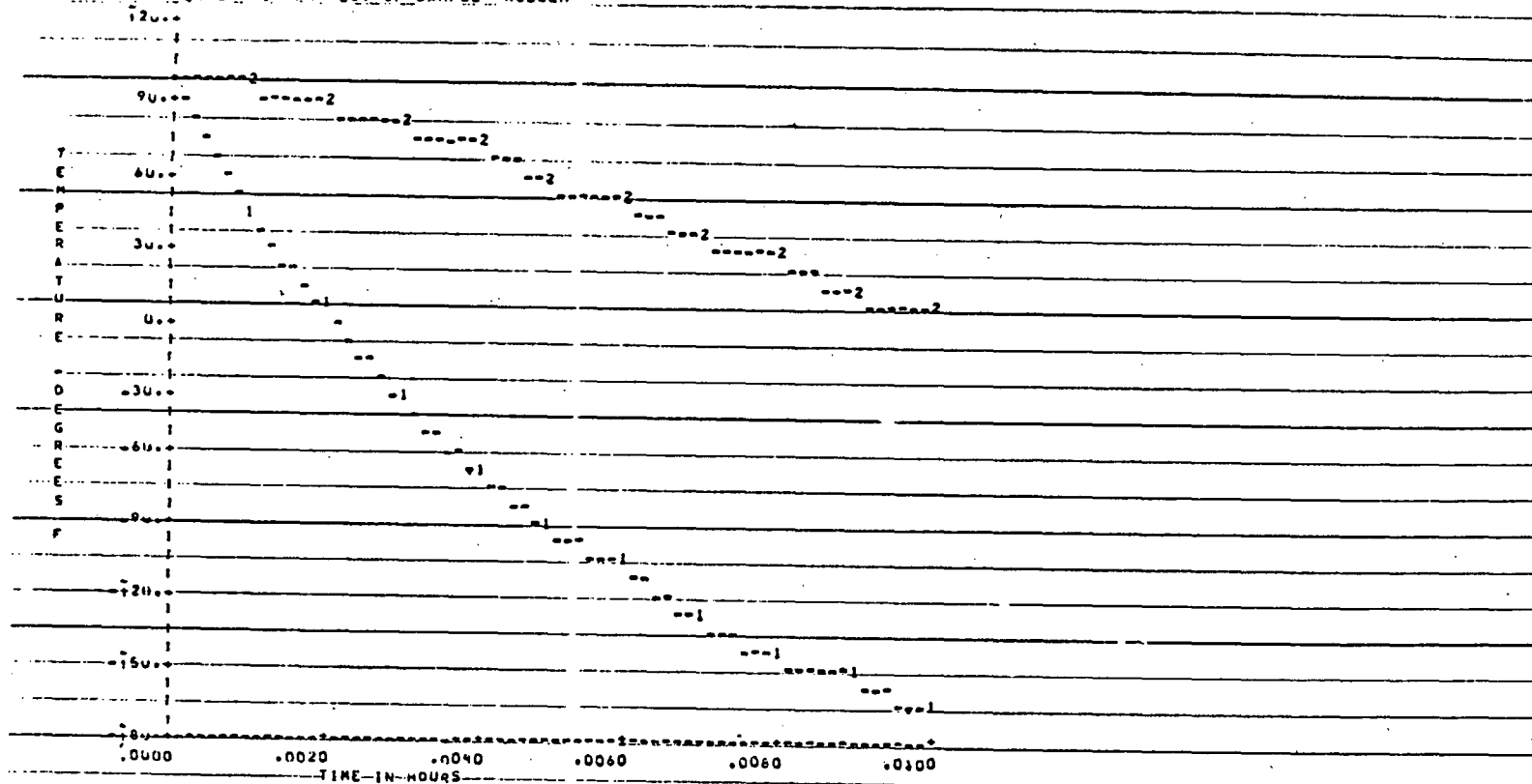
(h) Partial view of the interface ring.

Figure 8.- Concluded.

HYDRAULIC ACTUATOR FOR SSME MAIN ENGINE SAMPLE GRAPHICS PROBLEM

```
*****
TIME= 1.00000-02 DTIMEU= 5.38633-05 CSGHINI 20051= 1.04359-04 TEMPCCI 100031= 5.81692-01 RELXCCI 01= 0.00000
T 1001= 9.94835+01 T 1002= 9.94282+01 T 1003= 9.94282+01 T 1004= 9.94299+01 T 1005= 9.94299+01 T 1006= 9.94132+01
T 1007= 9.94132+01 T 1008= 9.94132+01 T 1009= 9.94132+01 T 2001= 9.97057+01 T 2002= 9.97080+01 T 2003= 9.97008+01
T 2004= 9.97333+01 T 2005= 9.97324+01 T 3001= 9.98782+01 T 3002= 9.98784+01 T 3003= 9.99465+01 T 3004= 9.99465+01
T 3005= 9.99766+01 T 3006= 9.99766+01 T 3007= 9.99877+01 T 3008= 9.99877+01 T 10001= 1.63227+02 T 10002= 1.64969+02
T 10003= 1.62429+02 T 10004= 1.64329+02 T 10101= 4.38727+00 T 10102= 4.55406+00 T 10103= 4.21714+00 T 10104= 4.27971+00
T 10201= 8.66449+01 T 10202= 8.59222+01 T 10203= 8.62353+01 T 10204= 8.23450+01 T 10301= 8.24923+01 T 10302= 8.23224+01
T 10303= 9.91826+01 T 10304= 9.84426+01 T 10401= 9.94782+01 T 10402= 9.90629+01 T 10403= 9.99069+01 T 10404= 9.95150+01
T 9999= 4.23000-02
```

LINE PRINTER PLOT OF SAMPLE PROBLEM



END OF DATA

Figure 9.- Sample SINDA output and line printer plots which can be viewed on the CRT.



**Page Intentionally Left Blank**

## QUICK-GEOMETRY

### A RAPID RESPONSE METHOD FOR MATHEMATICALLY

### MODELING CONFIGURATION GEOMETRY

Alfred F. Vachris, Jr., and Larry S. Yaeger

Grumman Aerospace Corporation

## SUMMARY

This paper outlines the philosophy, the development and various applications of the QUICK-GEOMETRY system. This system provides a practical method for developing the geometry models that are essential to the operation of computer-based design and manufacturing systems. QUICK-GEOMETRY is part of Grumman's Rapid Aerospace Vehicle Evaluation System (RAVES).

## NEED FOR GEOMETRY MODELING

Today, many efforts are focused on the design and operation of computerized systems for the development and production of all types of vehicles (aircraft, ships, automobiles and spacecraft). The results of these efforts are represented by computer-aided-design systems like ODIN, RAVES and IPAD (refs. 1, 2, and 3) and by computer-aided-manufacturing systems like CADAM (ref. 4).

The ability to model geometry is essential to the operation of these systems. From the systems standpoint, vehicle geometry must be modeled to establish a data base for geometry information. From the design standpoint, geometry must be modeled to develop the parametric models for conceptual design studies (ref. 5). From the analysis standpoint, configuration geometry must be carefully modeled to provide inputs for analysis programs (ref. 6). From the manufacturing standpoint, parts must be programmed so that they can be produced by numerically controlled machinery. In addition, surface geometry must be lofted to produce manufacturing templates (ref. 7).

## RECENT EVOLUTION IN THE METHODS FOR MODELING GEOMETRY

A complete geometry modeling system would support the variety of geometry needs that are listed above. Today, there are a number of different geometry systems that cover some aspects of the problem (see, for example, refs. 8, 9, and 10) but no system is complete. The QUICK-GEOMETRY system was developed as the initial step toward a total geometry system.

The best way to describe the QUICK-GEOMETRY system is to contrast this system with the other systems now in use. Of particular interest, are the various methods for modeling surface geometry that are used by aerodynamic analysis codes. These different methods can be cataloged according to the various data structures that are used to define surface geometry (see fig. 1). In cataloging the various codes it was convenient to split the geometry modeling into two parts: (1) A description of the geometry of cross-section cuts at select fuselage stations and (2) a description of the surface geometry between cross-section cuts.

### Point Geometry

The initial method for modeling geometry used a point geometry data structure that defined the configuration by tables of surface coordinates (refs. 11 through 14). Each table of surface coordinates is defined by digitizing a cross-section cut at a selected fuselage station. There are a number of difficulties with this approach: (1) Since a large number of surface points are required to develop this type of model, the process of measuring, cataloging and checking these surface coordinates becomes a very tedious and time consuming task. (2) Surface geometry is approximated by a large number of flat panels that are generated by connecting surface points from adjacent cross sections. The resulting surface is quite angular, making interpolation very difficult and providing only rough estimates for surface slopes and normals. (3) It is difficult to respond to design changes. Even local design changes require a careful review of how the original surface geometry was broken down into tables of points before the new geometry can be modeled. In many cases it is easier to remodel entire cross sections from the new drawing rather than modifying the earlier model. (4) The formats of point geometry models are very specific so that different analysis codes often require completely new models.

### Fixed Pattern Geometry

Of all the problems mentioned, the difficulty with surface interpolation is the most critical one. To get around this problem a number of improvements were made to the point geometry method. To begin with, a rule or a pattern was followed when digitizing cross-section cuts so that a proper framework for surface interpolation could be established by sequentially connecting surface points from adjacent cross sections. Next, the data structure was expanded to provide for the definition of a number of different patterns and to allow the vehicle to be broken up into a series of modules, such that each module contains all the cross sections that could be fit by the same pattern. Finally, the digitizing pattern, which was built up from segments of lines, circles and ellipses, was developed analytically. Once the pattern had been established it was a simple matter to digitize the necessary points from the plan and profile views. Digitizing from body lines rather than from the cross-section cuts that are scattered all over the drawing generates a more consistent model.

The sum total of these improvements produced a new modeling technique centered on the use of a pattern rather than points to define cross sections (refs. 15 and 16). However, like the point geometry method, these patterns are still defined only at specific fuselage stations so that surface geometry must be defined by interpolation. And although surface interpolation can be carried out, there are problems because the interpolation techniques (spline fit and least squares fit) are not an integral part of the pattern.

### Moving Pattern Geometry

Again, difficulties with defining surface geometry by interpolation lead to the next development in modeling. Instead of trying to improve interpolation techniques, a more comprehensive modeling system was developed. This system eliminated the need for surface interpolation by providing for the continuous definition of cross-section patterns along the length of the vehicle. In this system all the lines in the plan and profile view are individually modeled by either analytical or numerical (spline fit) techniques. The data structure then completes the model by analytically joining the cross-sections patterns to the body lines. In this way a cross-section pattern can be developed at each fuselage station (see refs. 17, 18 and 19).

The moving pattern technique seems like the correct way to model surface geometry. But the method must be organized into a proper geometry system before it can be evaluated. Since this method resulted from ad hoc modeling techniques to support aerodynamic computations, the codes that use this method have not set up geometry modeling systems.

The QUICK-GEOMETRY system will make an initial step toward the development of a total geometry system by organizing and developing the moving pattern into a proper geometry system.

### OBJECTIVES

The overall objective is to develop the moving pattern into a geometry modeling system that could service the complete spectrum of geometry needs to support design, analysis, manufacturing and CAD-CAM.

To make sure that the QUICK-GEOMETRY system could be used as a design tool, a number of specific objectives were identified:

- The method should be capable of generating geometry models at all levels of configuration definition, starting with the initial "back of the envelope" sketch through the final three-view drawing. Then modeling could begin while the design was still taking shape, instead of having to wait until the three-view drawing was complete.

- The method must be capable of adding new detail to the current model as the design is developed. Then the model could be continually updated and so avoid having to remodel for each new phase of development.
- The method should be capable of synthesizing conceptual design models by selecting vehicle components from cataloged shapes and then scaling, rotating and translating them until they are properly positioned on the model. With this capability the initial three-view could then become a by-product of the model.

A second group of objectives were identified to make sure that the system could function as part of a CAD-CAM data bank. These objectives were:

- Common geometry - The model must serve as a central source of input geometry for a number of programs.
- Complete geometry - The model must integrate the various configuration components so that the vehicle is handled as a single unit.
- Computational geometry - The model must be defined by mathematical equations so that the surface coordinates and derivatives are calculated analytically.

#### GROUND RULES

Realizing that the development of a new geometry modeling system would require a significant effort, a number of ground rules were set up to organize the development of the computer code.

The primary ground rule was that the modeling system should be split into two separate operations. The first phase of the split operation is to define the geometry from the drawing and to collect all the modeling information into a concise model that could then be stored in a data bank. The second phase of the operation is to look up (calculate) surface geometry using the model defined in phase one. The overall organization of the split operation is shown in figure 2. Phase one is called QUICKDEF, while phase two is called QUICKLOK.

Because of the operational split the codes could then be developed and optimized for the separate functions to be performed. The QUICKDEF code must be very flexible to do a proper modeling job and so it requires a number of user-oriented conveniences to help make the program easy to use. These conveniences were built into the QUICKDEF code by first designing a self-explanatory data deck that could handle the job of modeling a difficult configuration, and then developing the subroutines to process and organize this input deck into a concise mathematical model. The definition code uses the cartesian coordinate system shown in figure 3.

On the other hand, the QUICKLOK code must be very automatic so that the surface geometry can be looked up without requiring that the user have specific knowledge of how a particular configuration has been modeled. The code must be very efficient so that it can be used frequently without a cost penalty. The look-up model uses a cylindrical coordinate system which properly supports the moving pattern geometry (fig. 3). The QUICKLOK code was designed around the use of two subroutines that were to provide the communication between the user's program and the math model.

The two subroutines are GEOMIN and CSGEOM. The first subroutine reads the math model into core and sets up the data structure.

To read the model:

```
CALL GEOMIN (IREAD, IRITE, ICRITE)
```

Where: IREAD - input unit

IRITE - write unit

ICRITE - write unit for error messages

The second subroutine is called each time a surface point is to be calculated. To calculate a surface point:

```
CALL CSGEOM (X, H, R, RX, RH, RXX, RXH, RHH, NDERV)
```

Where: X - fuselage station location (see fig. 3)

H - theta location ( $-\pi/2 \leq \theta \leq +\pi/2$ )

R - radial distance to surface point (X, H)

RX -  $DR/DX$  at the surface point

RH -  $DR/D\theta$  at the surface point

RXX -  $D^2R/DX^2$  at the surface point

RXH -  $D^2R/(DXD\theta)$  at the surface point

RHH -  $D^2R/D\theta^2$  at the surface point

NDERV -  $\pm N$ , where N is the order of the derivative to be calculated (0, 1, 2).

+ N, previous call to CSGEOM was at a different surface point. Set up the geometry at this station before computing derivatives.

- N, previous call to CSGEOM was at the same surface point. Calculate derivative without recalculating R.

The quantities X, H, and NDERV are specified by the user, the remaining values are computed by the QUICKLOK code.

As a result of the split operation a number of other codes have been added to the QUICK-GEOMETRY system. They include:

- QUICKCHK - operates the QUICKLOK code to check out the model. Outputs are tables and plots of surface coordinates and slopes.
- QUICKPLT - a general plotting package that will display the model geometry from different points of view and in different display formats.
- QUICKGEN - generates point geometry input models for other computer programs.

To get the modeling techniques started, a number of limitations were put on the geometry model. First of all, only the external surface of the configuration will be modeled. All the internal geometry is ignored. Secondly, the surfaces will be evaluated in polar coordinates so that the surfaces must therefore be expressed as a single valued function of theta. A moving axis, in the plane of symmetry, is provided to lessen the impact of this limitation (see fig. 4). And finally, the basic modeling elements that are used to build up the geometry of the configuration make use of only point and slope boundary conditions; hence, curvature is not matched from one element to another. (Note - Even with these intentional limitations, experience has shown that the QUICK-GEOMETRY system does a very good job of modeling geometry.)

To make sure that the system was not dependent on a particular hardware system, two other ground rules were adopted. First, the modeling system was developed so that it could operate without the use of any special equipment such as digitizers or scopes. Of course, the performance of the system will be improved by the availability of extra hardware. The final ground rule was that the code be developed in machine-independent FORTRAN code so that it could be put up on any system with a minimum of effort. There are no machine tricks in the QUICK-GEOMETRY system. Furthermore, the code was developed on IBM equipment using single precision for the most part, so there will be no precision problems when running on other computer systems. As a result of these ground rules, the use of the QUICK-GEOMETRY system is not limited to organizations having access to large-scale computer facilities.

## UNIQUE FEATURES

The QUICK-GEOMETRY system supports a number of unique features that distinguish this geometry system from all others. These features include the data structure, a user-oriented language for geometry modeling and the formulation of a geometry catalog.

## DATA STRUCTURE

The most important feature of the QUICK-GEOMETRY system is the data structure. The left-hand side of figure 5 shows the different levels of organization that are used in the data structure. The pattern or surface which has the most complex organization is at the top, leading down to the curve element which is the fundamental building block of the QUICK-GEOMETRY system. The chart in the center of the figure shows how the various parts of the data structure are used to model the configuration and it also provides a functional flow diagram for both the QUICKDEF and the QUICKLOK codes. The flow from top to bottom outlines the different steps taken to translate a drawing into a QUICK-GEOMETRY model, while the path from bottom to top outlines the different computational steps used to construct surface geometry from the math model.

### Curve Element

The basic building block of the QUICK-GEOMETRY system is the curve element (see fig. 6). The curve element is similar to the standard lofting conic except that the shoulder point condition has been replaced by a direct selection of a shape equation.

The boundary conditions that are used to determine the coefficients of each shape equation are the coordinates of the origin point, the termination point and the slope control point. The slope control point is a very convenient way of specifying slope conditions in coordinate form and in particular, it allows for the simultaneous specification of a vertical tangent at one end point and a horizontal tangent at the other.

Although there are a variety of shape functions to choose from, it is easy to make the proper selection. Aside from the line and the cubic, all the shapes fall into either the x-parabola or the y-parabola family. The x- or y-prefix indicates that the axis of symmetry for that particular parabola is parallel to the x-axis or y-axis. The x- or y-prefix for the rotated parabola and for the ellipse is very important because it identifies the shape that the curve element will take when the c-coefficient in the shape equation has a zero value.

The shape selection is made by matching the geometry to be modeled to either the x- or y-parabola. If the segment to be modeled has a vertical slope along its length, then it belongs to the x-family; if it has a horizontal tangent, then it belongs to the y-family. If the segment has both a vertical and a horizontal tangent, then it must be modeled as an ellipse. Note that the x-ellipse and y-ellipse generate the same points provided that  $c \neq 0$ . If the segment is rotated so that its symmetry line is not parallel to either the x-axis or the y-axis, the segment must be modeled as a rotated parabola. Note that for intermediate slope conditions, both rotated parabolas could be defined for the same set of conditions and that each shape would generate a different curve.



## Curves

The curve is the next level in the data structure. In the QUICK-GEOMETRY system each curve is the projection of a space curve into either the plan or profile view of a three-view drawing. Every curve is constructed by blending various curve elements together, with user control over matching point and slope continuity between segments (see fig. 7).

<u>TYPE</u>	<u>KEYWORD</u>	<u>FUNCTION</u>
Piece	PIECE	Curve is defined as a unit, with end points and slope control point if necessary.
Aft-Link	ALINK	Curve being defined begins at the end of the previous curve and is tangent to it.
Fore-Link	FLINK	Curve being defined ends at the beginning of the following curve and is tangent to it.
Patch	PATCH	Curve being defined begins at the end of the previous curve and ends at the beginning of the following curve and is tangent to both of the adjoining curves.
Fillet	FILET	End points and slopes of curve being defined are calculated from specified positions on the adjoining curves.

Note that the fillet, in addition to defining a new segment, also controls the domain or use of the curve segments that are adjacent to it.

## Surface Element

The next level in the data structure is the surface element which is very similar to the curve element. At a fixed fuselage station the surface element would be defined exactly like the curve element by specifying the coordinates for the origin, the termination and the slope control points. However, as shown in figure 8, the surface element is not fixed at a particular fuselage station, but rather, is continuously defined along some portion of the fuselage. The boundary conditions for the surface element are therefore defined along curves specifying the variation of the origin, the termination and the slope control points.

At each fuselage station the coordinates of the boundary curves are used to calculate the coefficients in the shape equation defining each surface element. Currently, the shape equations for the surface elements are limited to the line and the ellipse. Note, however, that  $A^2$  and  $B^2$  are used as coefficients in the ellipse equation, instead of just A and B, therefore the "ellipse" equation can also define a hyperbola.

Since the shape equation coefficients are calculated continuously along the length of the fuselage, these coefficients are functions of  $x$  just like the boundary curves ( $x$  is the distance along the fuselage reference line aft of some reference point). The derivatives of these coefficients can also be calculated from the boundary curves. Once these derivatives are known, then the surface shape function can be differentiated to produce the set of surface derivatives shown in figure 8.

## Surfaces

The surface or pattern is the top level of the QUICK-GEOMETRY data structure. The pattern is built up from surface elements in two ways: (1) using the same blending controls that were used to build up curves from curve elements (see fig. 7) and (2) using surface elements to control the use of adjacent surface elements.

The right-hand side of figure 9 illustrates the basic concept of the adapting or controlling surface element. The canopy and the fuselage are each defined as a separate surface element while the surface (canopy plus fuselage) is defined by specifying that the canopy and the fuselage are to control the use of each other. The basis of this control is the ability to calculate the intersections of adapting elements, and to then limit the use of each element, so that only external geometry is seen.

The left-hand side of the figure shows the fillet which is a special type of adapting surface element. The fillet is special because it is an optional surface element that can be added to a pattern that has already been modeled. It would be impractical to add a fillet to a pattern using only measurements from a drawing because the measurements cannot match the pattern coordinates which are computed to at least six significant figures. Adding tolerances to the program is not the answer because this will only lead to a tangled mess of logical tests.

A straightforward procedure was developed for adding a fillet. The fillet is defined by giving only one measured coordinate at each end point of the fillet (usually the  $y$ -coordinate at one end and the  $z$ -coordinate at the other). The program then calculates the missing coordinates from the pattern. Thus, the fillet will properly adhere to the pattern. For example, look at the fillet that is shown under the wing in figure 9. This fillet is defined by the  $z$ -coordinate of the inboard end point and by the  $y$ -coordinate of the outboard end point.

## MODELING LANGUAGE

The QUICKDEF code uses a user-oriented language for translating a drawing into a math model. The basic elements of this language are the keywords that define shapes and blending control and a common input structure that defines both surface and curve elements. Some of these language features are shown in a sample input deck (see fig. 10).

The first six cards after the title card define the single pattern that is used for this model. The pattern shown here has two surface elements: the lower body (BDYLOWER) and the upper body (BDYUPPER). Each surface element is defined by its shape keyword, its blending keyword and the names of its origin, termination and slope control curves. All the names (excluding the shape and blending keywords) are chosen by the user. Selecting functional or mnemonic names will make it easier to follow and model the various lines shown on a three-view drawing.

The next portion of the data deck (starting with the card showing YBDYBOT) models the plan and profile views of the lines that define the surface elements. The particular curve to be modeled is identified by prefixing a y (for plan view) or a z (for profile view) to the name of the control line. The next two cards define one of the curve elements that will build up the curve. The first card specifies the keywords for shape and blending. The second card defines the coordinates for the origin, termination and the slope control points. Depending on the blending control, this data may be provided by pointing to another curve element rather than typing in the coordinate values. A card with a minus one signals the completion of input for the curve being modeled.

The last part of the data deck shows the alias cards which set up a way of sharing curve models instead of having to model the same curve twice. For example: the plan projection of the body top center line (YBDYTOP) and the plan projection of the body bottom center line (YBDYBOT) are the same line on the drawing so they are aliased to share the same curve model. Aliases are especially useful for defining the slope control lines for surface elements. These slope control lines usually specify either a horizontal or a vertical tangent boundary condition at one end point of the surface element. A horizontal tangent is set by aliasing the z-coordinate of the slope control line to the z-coordinate of the end point line and a vertical tangent is set by aliasing the y-coordinate of the slope control line to the y-coordinate of the end point line.

## GEOMETRY CATALOG

The flexibility of the QUICK-GEOMETRY data structure and the modeling language makes it possible to collect and catalog the various patterns to form a library of configurations.

Figure 11 illustrates the flexibility of the pattern. This is the most elementary pattern consisting of only two elliptical surface elements. Once the pattern has been defined the actual shape that it will take at any fuselage station depends upon the relative locations of the boundary curves. All the shapes that are displayed come from the same pattern.

Figure 12 shows how the pattern can have a flexible structure, as well as a flexible shape. The adapting surface elements can be joined to build an adapting pattern that will change its external structure depending upon what components are visible. This can be accomplished, for example, by defining the canopy and the wing to be inside the fuselage at stations before they actually appear on the configuration and setting up the mutual use-domain controls among the fuselage, canopy and wing. Then the structure of the pattern will change automatically as canopy and wing appear and then disappear.

Selecting the correct patterns from the library is easy to do since the pattern is described completely in words.

## APPLICATIONS

The QUICK-GEOMETRY system is being applied in a number of ways. As part of RAVES it is used as a common source of vehicle geometry which is drawn upon to generate a variety of geometry input data decks (ref. 20). In particular, data decks conforming to the Harris input format (ref. 11), which is a standard at NASA Langley, can be generated. In another RAVES application, QUICK-GEOMETRY has been adapted to a vehicle lofting system (ref. 21). The most sophisticated application to date, has been to integrate the QUICK-GEOMETRY system with a numerical flow code which calculates the steady super/hypersonic inviscid flow around real configurations (ref. 22). In this application vehicle geometry (surface coordinates, slopes and normals) are generated as required by the flow analysis code. A small cross section of models are shown in figure 13.

## CONCLUSIONS

The QUICK-GEOMETRY system has developed the moving pattern method into a practical method for modeling geometry. It therefore provides the initial step toward a complete modeling system that can support design, analysis, manufacturing and CAD-CAM.

## WHAT IS NEXT ...

The data structure will be expanded:

- (1) To model multiple external and internal surfaces.
- (2) To define wings, fins and tails using natural aerodynamic coordinates (buttline and waterline cuts).
- (3) To define body lines by word patterns.

The design synthesis capability will be developed. Figure 14 shows the early results of a design effort based on the lofting program (ref. 21). The model shown was composed in a half hour using an IBM 2250.

And finally, an interface will be developed to join a surface patch technique to the moving pattern.

#### REFERENCES

1. Glatt, C. R., and Hague, D. S. : ODIN - Optimal Design Integration System. NASA CR-2492, February 1975.
2. Wennagel, G., Loshigian, H., and Rosenbaum, J.: RAVES - Rapid Aerospace Vehicle Evaluation System. ASME Winter Annual Meeting, Houston, Texas, 1975.
3. Heldenfels, R. R.: Integrated Computer-Aided Design of Aircraft. AGARD Conference Proceedings, No. 147, Vol. 1, 1973.
4. Feder, Aaron : Test Results on Computer Graphics Productivity for Aircraft Design and Fabrication. AIAA Paper 75-967, 1975.
5. NATO, Aircraft Design Integration and Optimization. AGARD Conference Proceedings, No. 147, Vol 1, 1973.
6. Giles, Gary L., Blackburn, Charles L., and Dixon, Sidney C.: Automated Procedures for Sizing Aerospace Vehicle Structures (SAVES). Journal of Aircraft, Vol 9, No. 12, December 1972, pp 812-819.
7. Bezier, P.: Numerical Control Mathematics and Applications. John Wiley and Sons, New York. 1970.
8. Forrest, A. R. : Computational Geometry Proceedings of the Royal Society of London, Vol 321A, pp 187-195, 1971.
9. Sutherland, I. E. : Three-Dimensional Data Input by Tablet. Proceedings of the IEEE, Vol 62, pp 453-461, April 1974.
10. Ahuja, D. V., and Coons, S. A.,: Geometry for Construction and Display. IBM System Journal, Vol 7, No. 3 and 4, 1968.
11. Craidon, Charlotte B. : Description of a Digital Computer Program for Airplane Configuration Plots. NASA TM X-2074, 1970.
12. Harris, Roy V., Jr.: An Analysis and Correlation of Aircraft Wave Drag. NASA TM X-947, 1964.

13. Hess, J. L. and Smith, A. M. O.: Calculation of Non-Lifting Potential Flow About Arbitrary Three-Dimensional Bodies. Douglas Aircraft Company, Rept. No. E.S. 40622, March 1962.
14. Woodward, F. A. : An Improved Method for the Aerodynamic Analysis of Wing-Body-Tail Configurations in Subsonic and Supersonic Flow. NASA CR-2228, May 1973.
15. Gentry, A. E., Smyth, D. N., and Oliver, W. R.: The Mark IV Supersonic-Hypersonic Arbitrary-Body Program. AFFDL-TR-73-159, Vol 1, November 1973.
16. Watman, H. and Meyer, R.: Grumman High Speed Aerodynamic Prediction Program (GAC-HAPP). Grumman Advanced Development Report ADR 01-03-74.3, March 1974.
17. De Jarnette, Fred R.: Surface Fitting Three-Dimensional Bodies. NASA CR-139663, 1974.
18. Gunness, R. C., Jr., Knight, C. J., D'Sylva, E.: Flowfield Analysis of Aircraft Configurations Using a Numerical Solution to the Three-Dimensional Unified Supersonic/Hypersonic Small-Disturbance Equations. NASA CR-1926, February 1972.
19. Kutler, P., Reinhardt, W. A., and Warming, R. R.: Multishocked, Three-Dimensional Supersonic Flowfields With Real Gas Effects. AIAA Journal, Vol 11, No. 5, pp 657-664, May 1973.
20. Vachris, A. : QUICK-GEOMETRY (A9), Grumman Aerospace Corporation. EG/RAVES-UM-210-75, July 1975.
21. Abramson, W. : Fuselage Modeling Program (C3), Grumman Aerospace Corporation. EG/RAVES-UM-102, July 1975.
22. Marconi, F., Yaeger, L., and Hamilton, H.: Computation of High-Speed Inviscid Flows About Real Configurations. NASA SP-347 (Part II), March 1975, pp 1412-1455.

DATA STRUCTURE	DEFINITION	CROSS-SECTION GEOMETRY	SURFACE GEOMETRY	COMPUTER CODES
POINT GEOMETRY	DIGITIZE CROSS SECTIONS	NUMERICAL APPROXIMATION	NUMERICAL APPROXIMATION	CRAIDON HARRIS HESS-SMITH WOODWARD
FIXED PATTERN GEOMETRY	DIGITIZE BODY LINES AT FIXED CROSS SECTIONS	ANALYTICAL COMPUTATION	NUMERICAL APPROXIMATION	GENTRY WATMAN
MOVING PATTERN GEOMETRY	DIGITIZE BODY LINES	ANALYTICAL COMPUTATION	ANALYTICAL COMPUTATION	DE JARNETTE GUNNESS KUTLER MARCONI

Figure 1.- Methods for defining geometry.

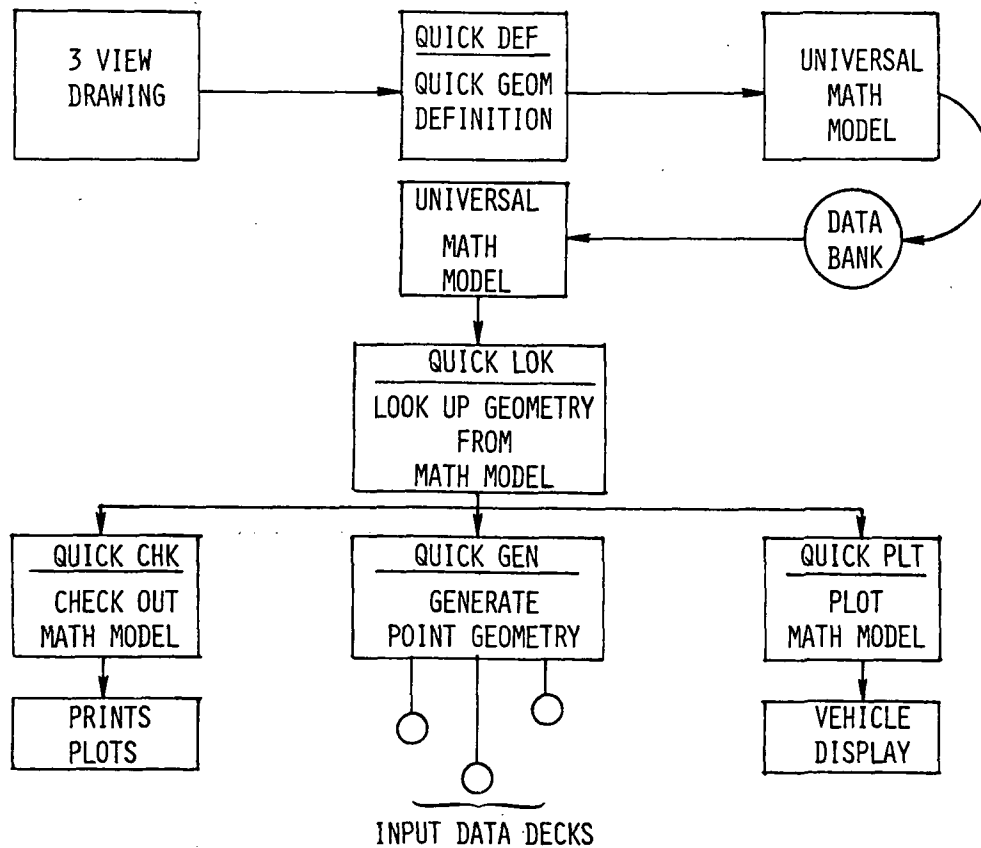


Figure 2.- Split operation.

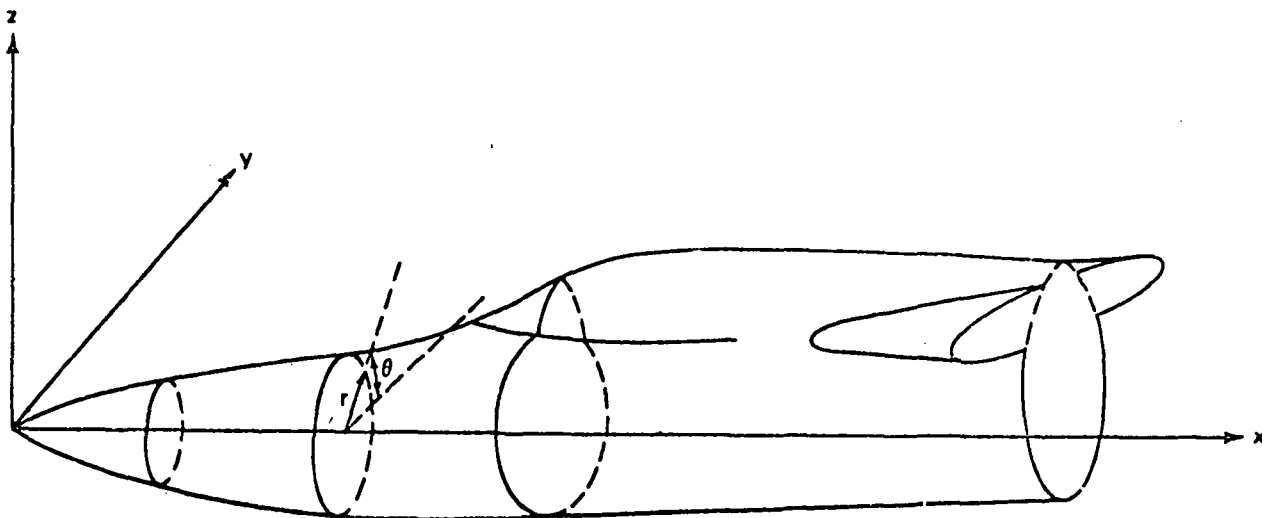


Figure 3.- Cartesian and cylindrical coordinate systems.

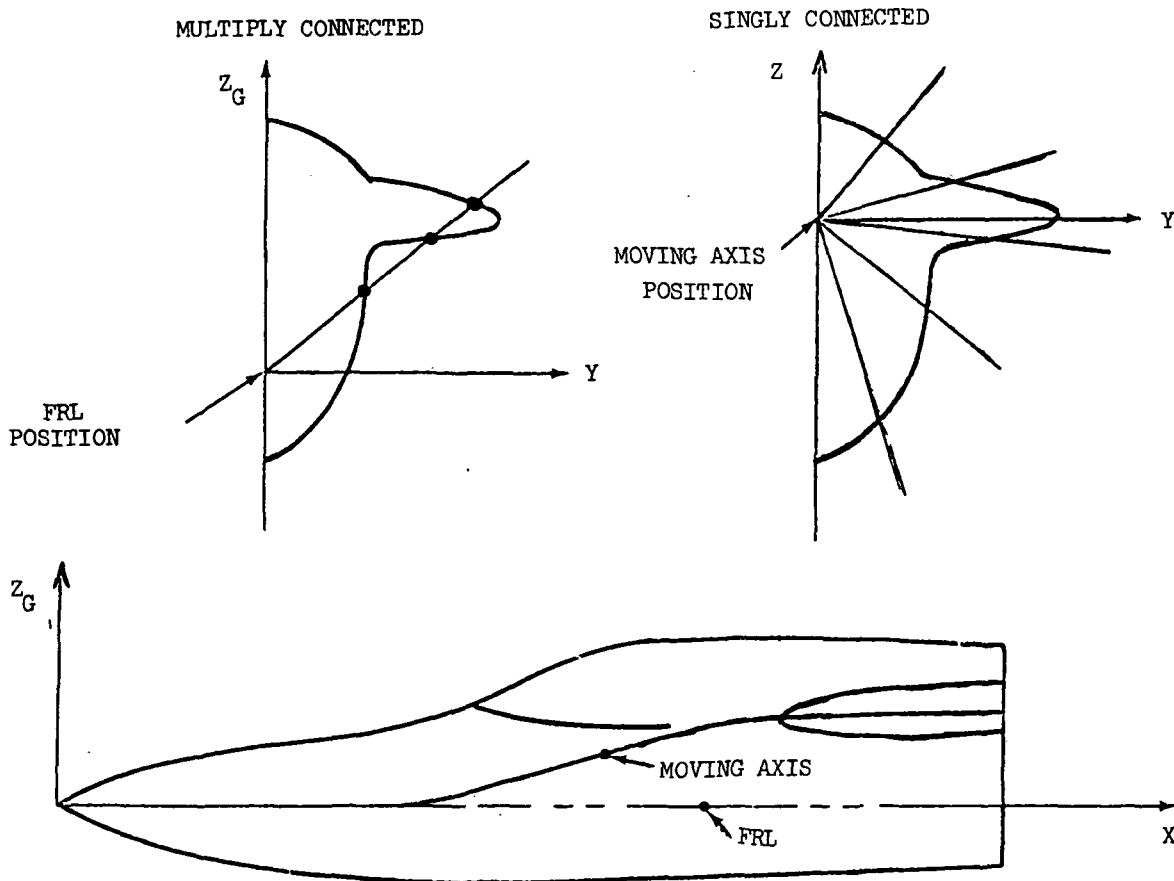


Figure 4.- FRL and moving axis.



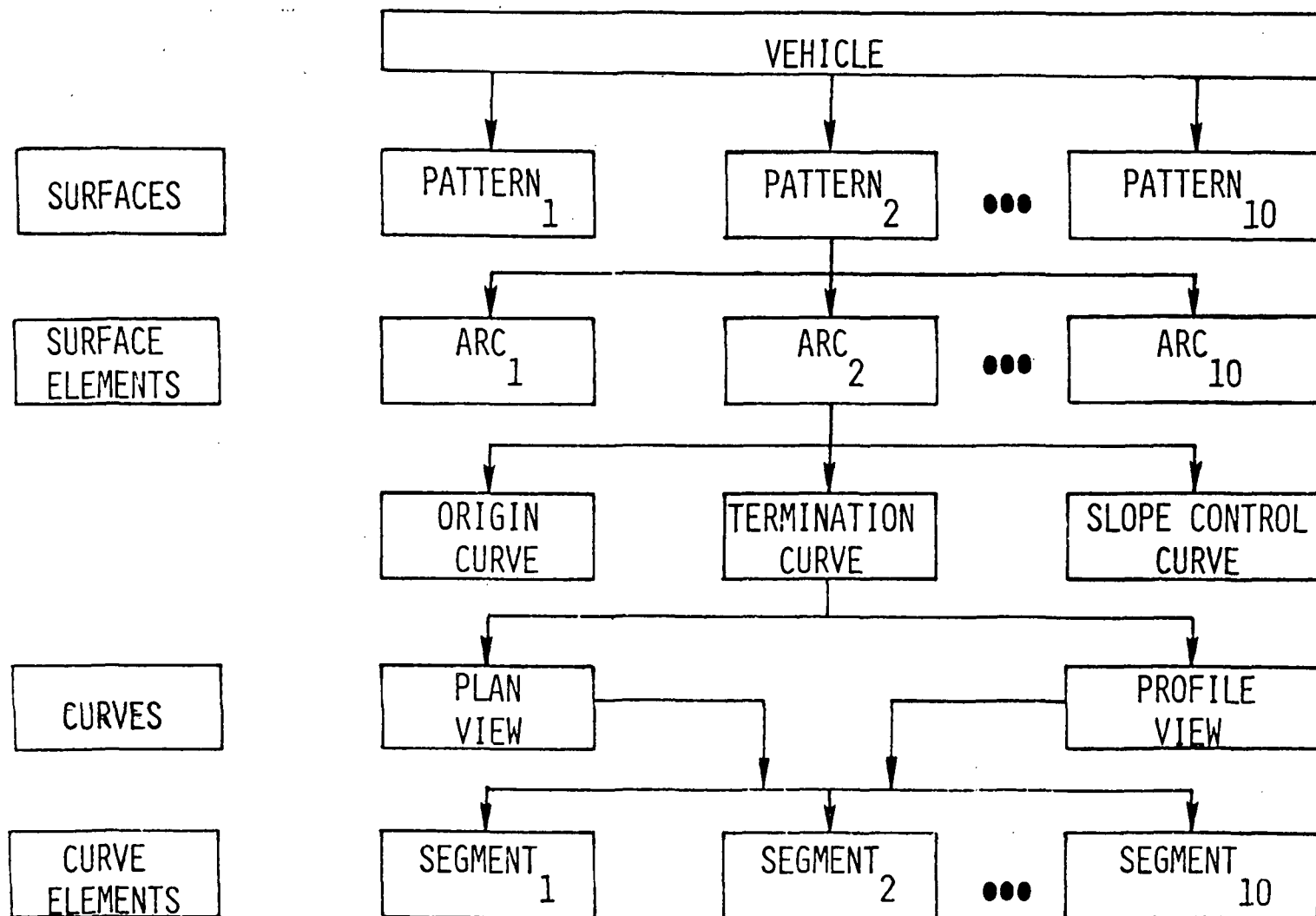
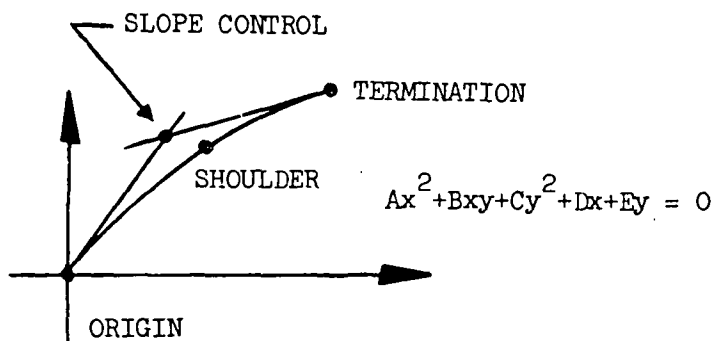
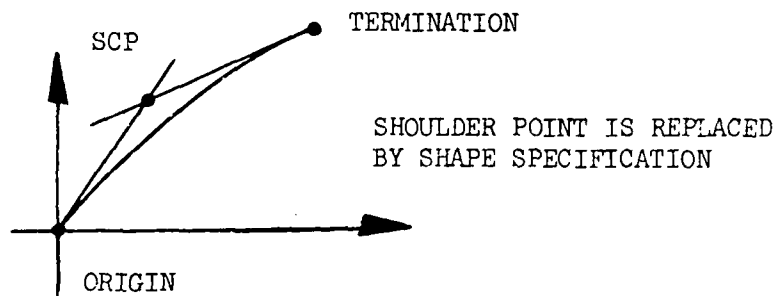


Figure 5.- Data structure.

## STANDARD LOFTING CONIC



## QUICK CURVE SEGMENT



SHAPE	KEYWORD	EQUATION	X-AXIS PARABOLA	ROTATED X-AXIS PARABOLA	X-ELLIPSE
Line	LINE	$Ax + By = 0$			
x-Parabola	XPAR	$Ax + By + y^2 = 0$			
y-Parabola	YPAR	$Ax + By + x^2 = 0$			
Rotated x-Parabola	RXPA	$Ax + By + Cxy + y^2 = 0$			
Rotated y-Parabola	RYPAR	$Ax + By + Cxy + x^2 = 0$			
x-Ellipse	ELLX	$Ax + By + Cx^2 + y^2 = 0$			
y-Ellipse	ELLY	$Ax + By + Cy^2 + x^2 = 0$			
Cubic	CUBI(C)	$Ax + By + Cx^2 + x^3 = 0$			
			Y-AXIS PARABOLA	ROTATED Y-AXIS PARABOLA	Y-ELLIPSE

Figure 6.- Curve element definition.

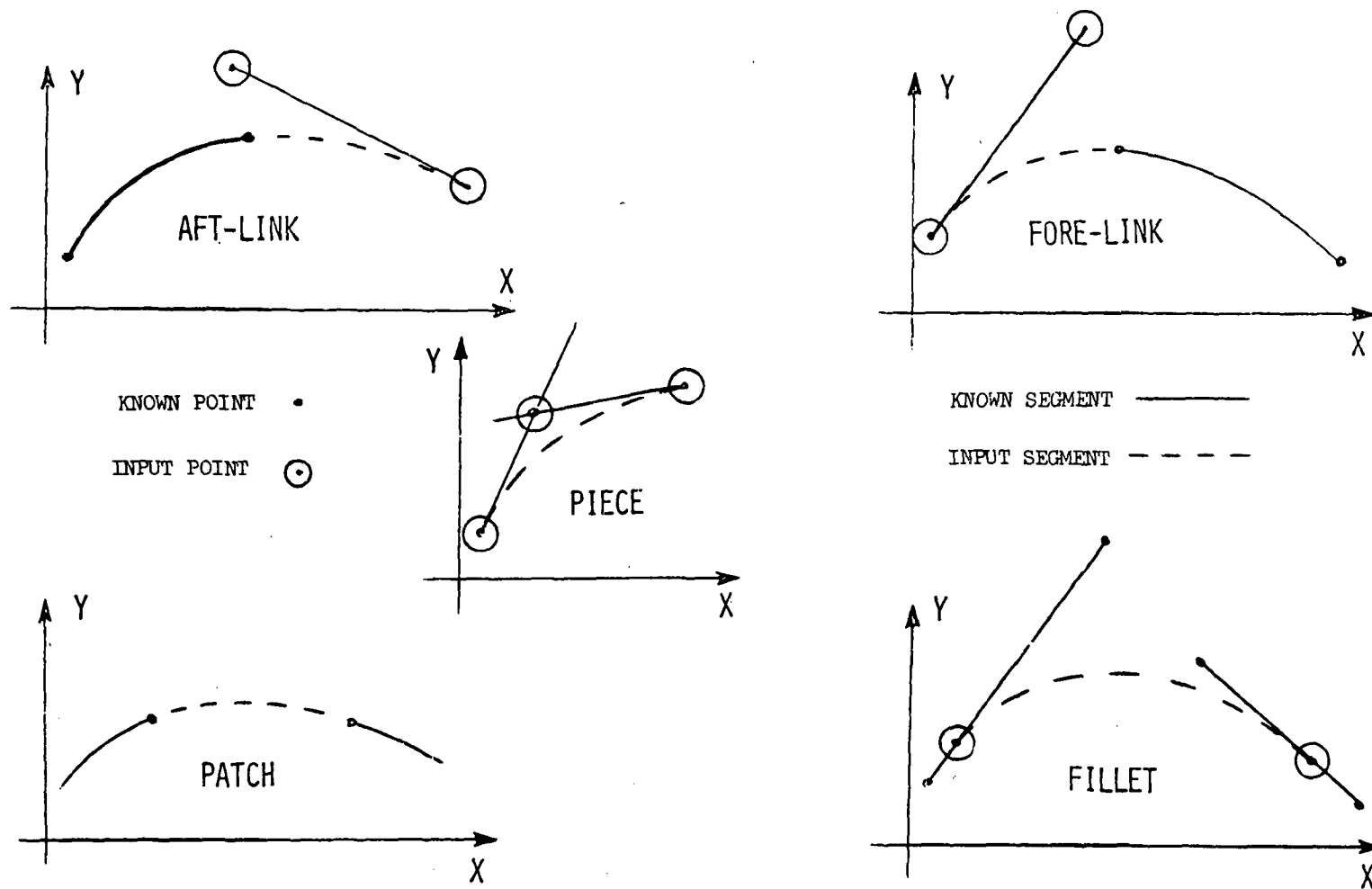
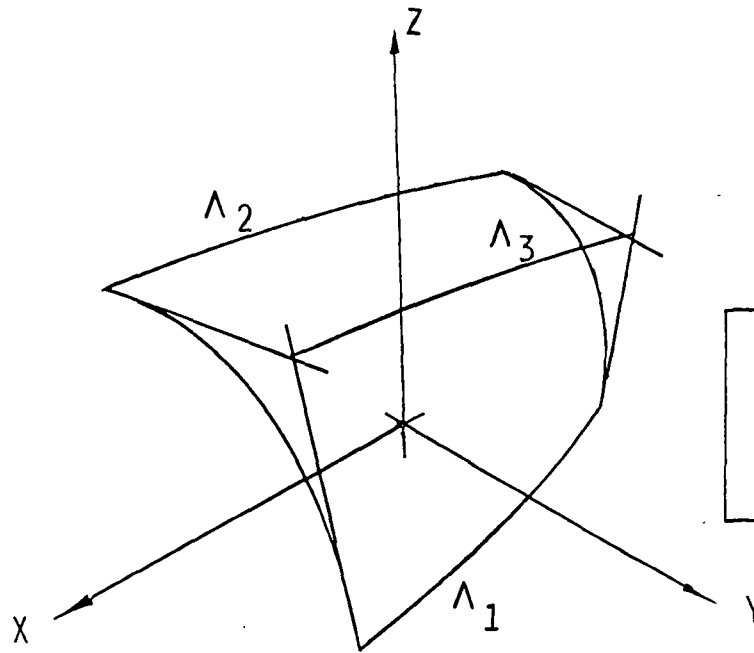


Figure 7.- Blending control for curve elements.



- $\Lambda_1$  - ORIGIN CURVE
- $\Lambda_2$  - TERMINATION CURVE
- $\Lambda_3$  - SLOPE CONTROL CURVE

TYPICAL SHAPE  $\rightarrow$  GROWING ELLIPSE

$$\frac{[Y(X) - Y_0(X)]^2}{A^2(X)} + \frac{[Z(X) - Z_0(X)]^2}{B^2(X)} = 1$$

POLAR FORM  $Q(R, R_0, \theta, \theta_0, A^2, B^2) = 0$

$$B^2 (R \cos \theta - R_0 \cos \theta_0)^2 + A^2 (R \sin \theta - R_0 \sin \theta_0)^2 - A^2 B^2 = 0$$

WHERE  $R_0 = R_0(X)$ ;  $\theta_0 = \theta_0(X)$ ;  $A^2 = A^2(X)$ ;  $B^2 = B^2(X)$

Q IS DIFFERENTIABLE PRODUCING

$$\frac{\partial R}{\partial X}, \frac{\partial R}{\partial \theta}, \frac{\partial^2 R}{\partial X^2}, \frac{\partial^2 R}{\partial X \partial \theta}, \frac{\partial^2 R}{\partial \theta^2}$$

Figure 8.- Surface element.

## ADAPTING SURFACE ELEMENTS

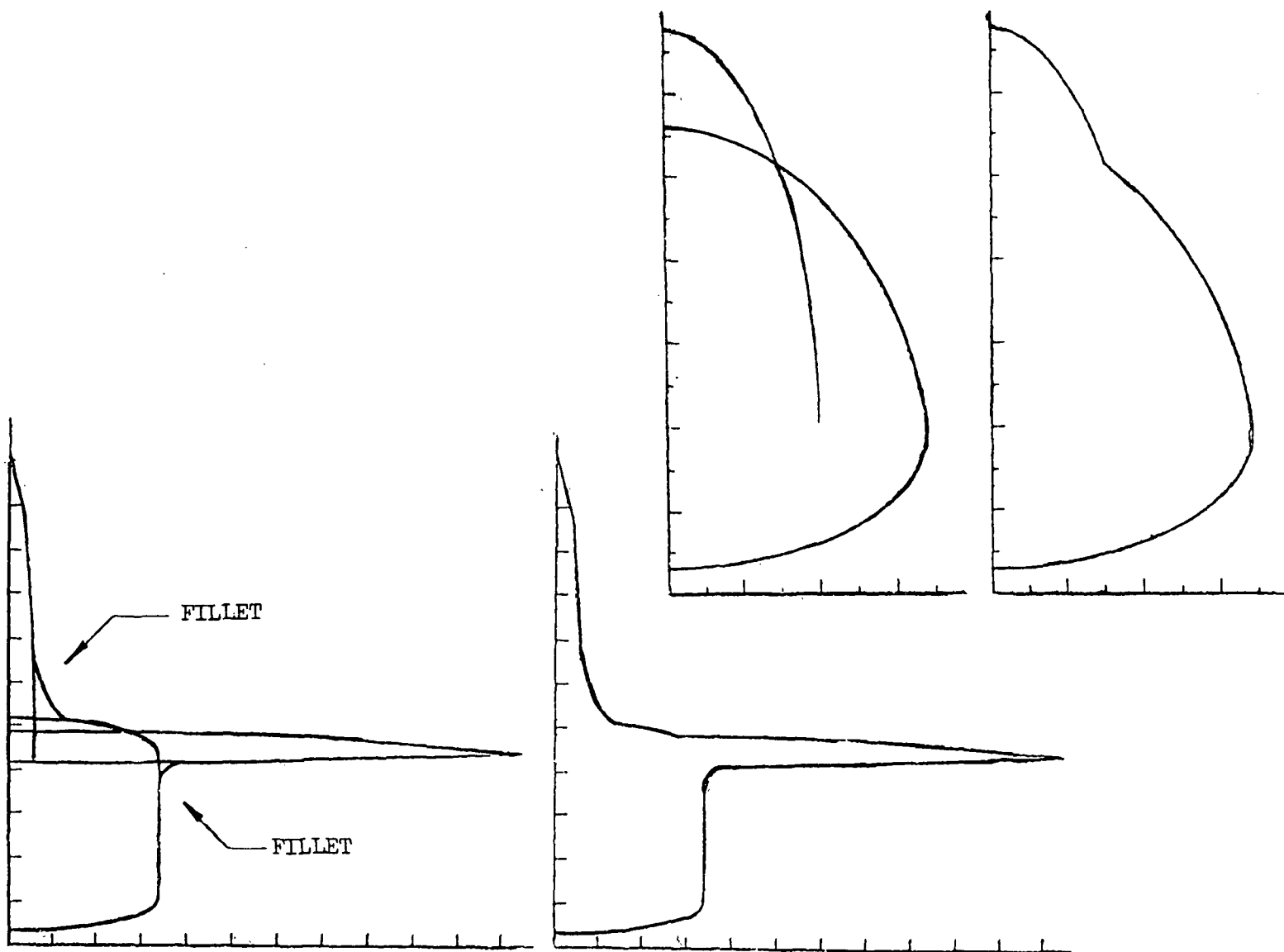
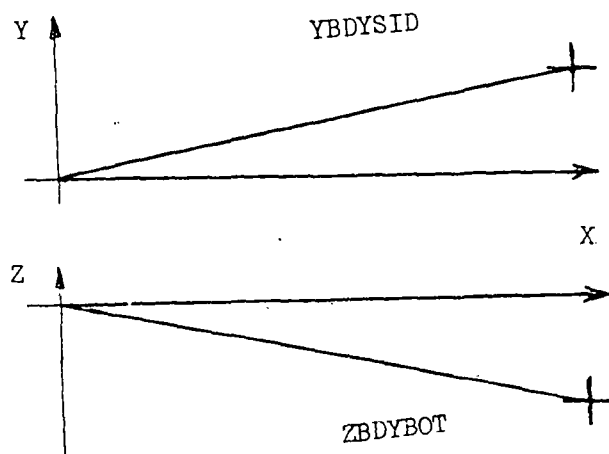
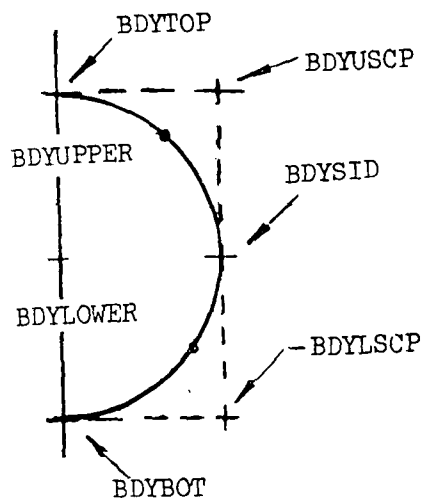


Figure 9.- Constructive geometry.



# SCONE20: TWENTY DEGREE SHARP CONE

```

1
1 2
BDYLOWER 1ELLI PIECE BDYBOT BDYSID BDYLSCP
BDYUPPER 2ELLI PIECE BDYSID BDYTOP BDYUSCP
1
1 1 MAPAXIS
0. 20.
YBDYBOT
1 LINE PIECE KV5
0. 20. 0.
-1
ZBDYBOT
1 LINE PIECE KV4
0. 15. A-20.
3 LINE PIECE KV5
10. -4.55 20. -4.55
2 RYPA FILET KVO
1. 3. 3. 10. 15.
-1
YBDYSID
1 LINE PIECE KV4
0. 15. A20.
3 LINE PIECE KV5
10. 4.55 20. 4.55
2 RYPA FILET KVO
1. 3. 3. 10. 15.
-1
ZBDYSID YBDYBOT
YBDYTOP YBDYBOT
ZBDYTOP YBDYSID
YBDYLSCP YBDYSID
ZBDYLSCP ZBDYBOT
YBDYUSCP YBDYSID
ZBDYUSCP ZBDYTOP
YMAPAXIS YBDYBOT
ZMAPAXIS YBDYBOT

```

Figure 10.- Sample input.

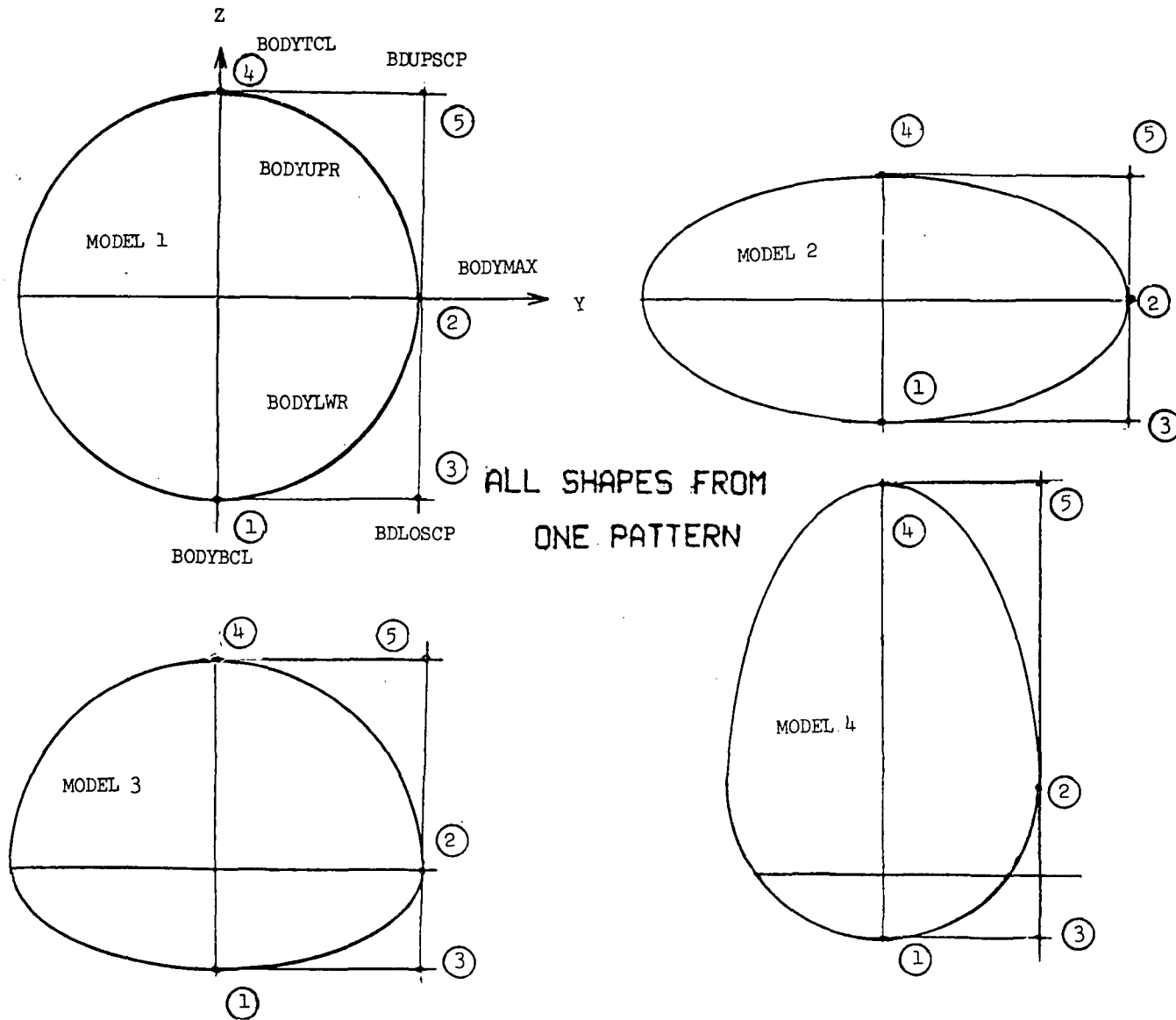


Figure 11.- Flexible pattern.

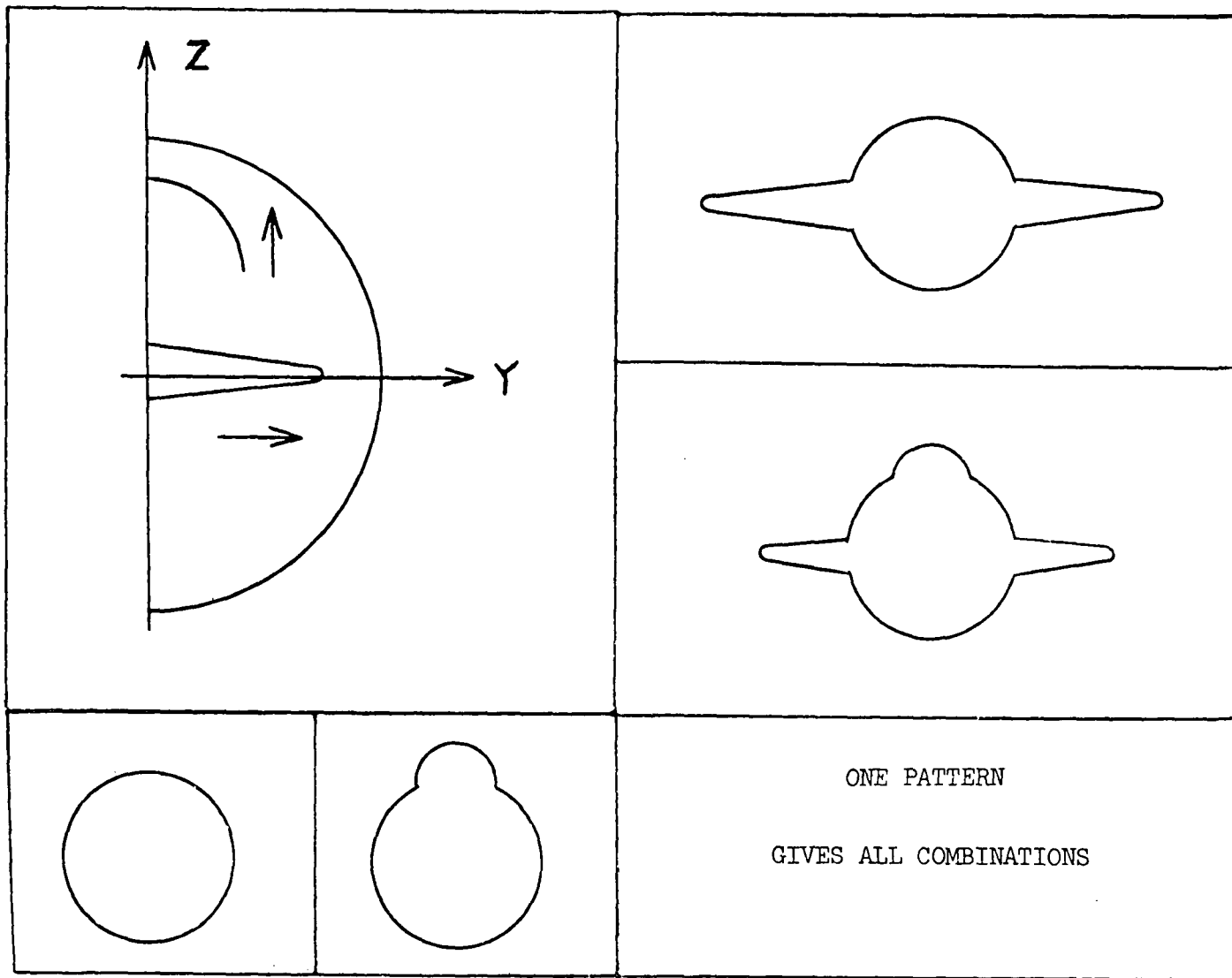


Figure 12.- Adapting pattern.



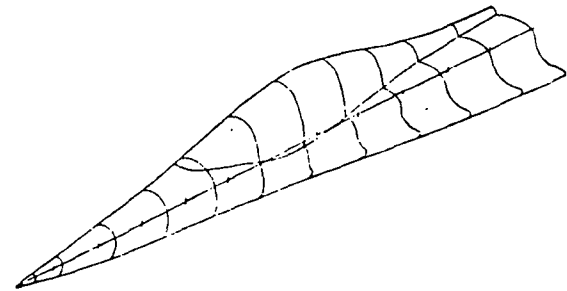
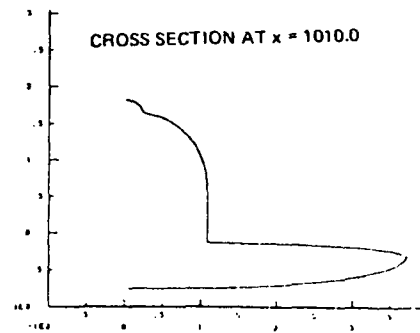
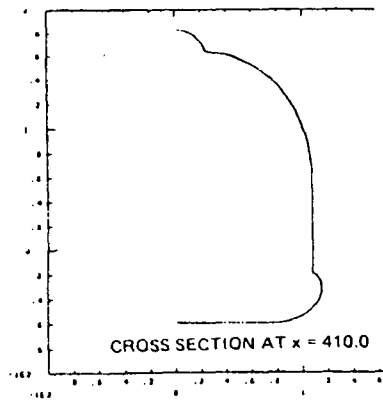
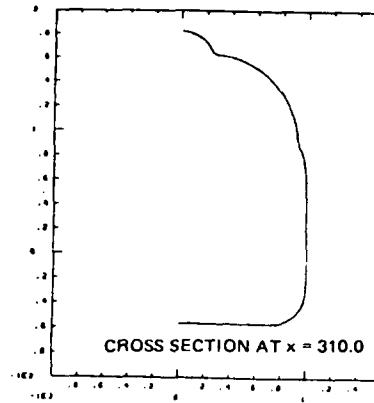
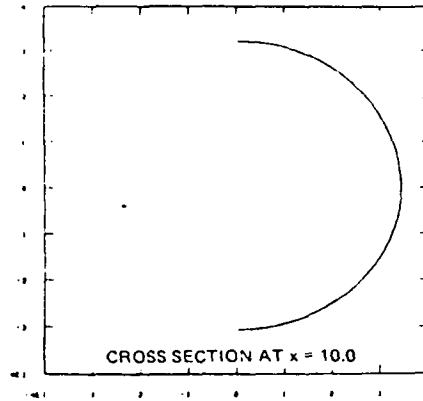
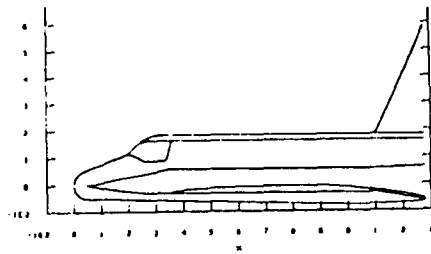


Figure 13.- Quick-geometry models.

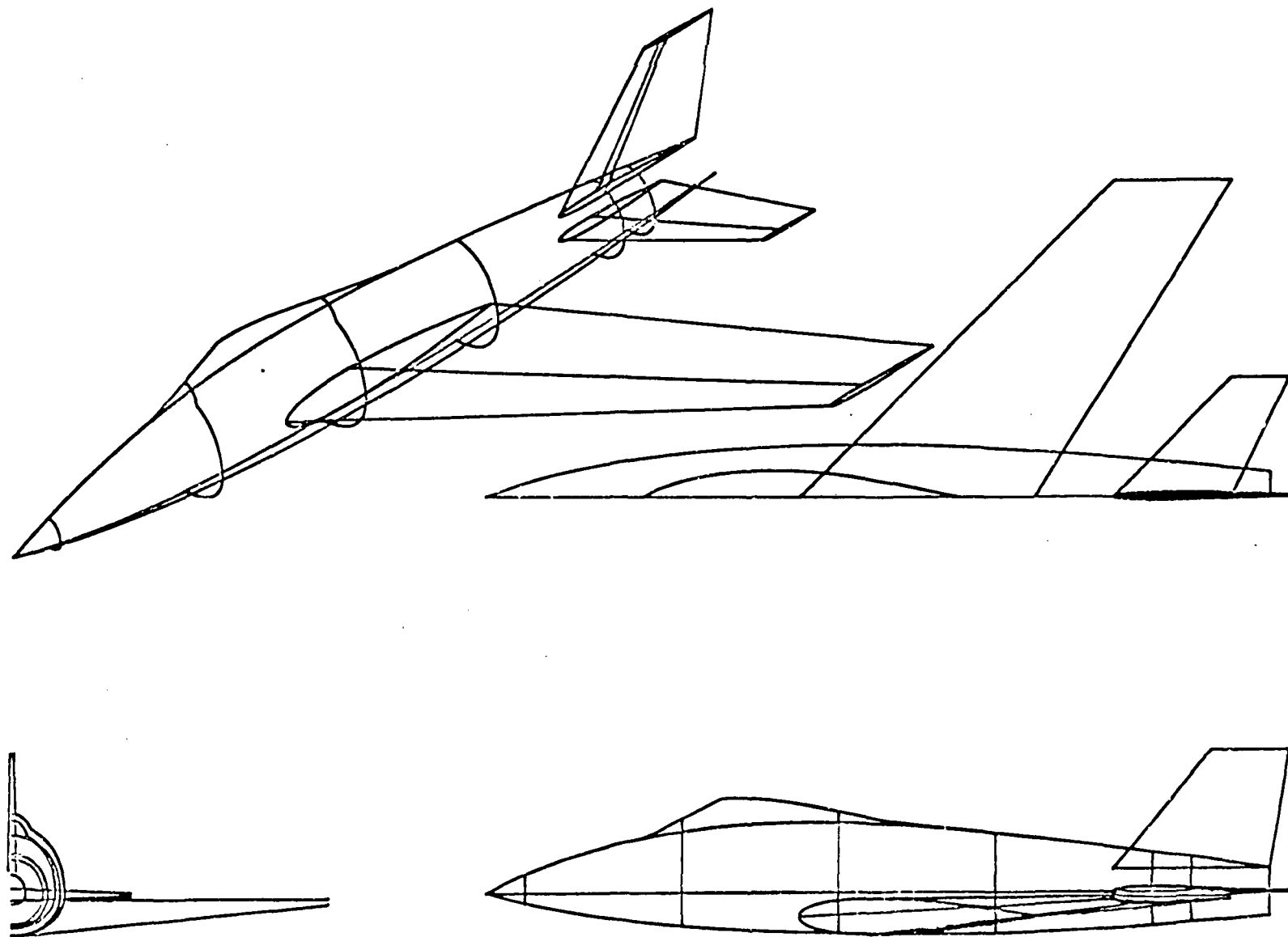


Figure 14.- Design synthesis.

**Page Intentionally Left Blank**

# COMPUTER-GENERATED ANIMATION FOR ANALYSIS AND DESIGN

Larry J. Feeser

Department of Civil Engineering  
Rensselaer Polytechnic Institute

## SUMMARY

Computer-generated animation and its application to engineering problems has received considerable attention recently. The development of computer-generated animation techniques is reviewed and some examples of the current state of the art are described and/or presented.

## INTRODUCTION

The area of computer-generated animation as a tool for engineering analysis and design has been under development for a considerable number of years. In this paper, an attempt is made to describe some of the salient features of the development of computer animation techniques. This is accomplished by examining some of the history of the development, by exploring the types of animation techniques in general use and by examining the motivation of individuals desiring to use animation in their engineering work.

A number of the ways in which computer-generated animation can be used will be examined in relationship to the suitability for the engineering task at hand. There have been developed a number of techniques for producing animation with computers and each of these will be explored and evaluated in terms of their suitability for use in engineering analysis and design. Finally, a number of the application areas in which computer graphics has been used in a beneficial manner will be described and illustrated.

## HISTORY AND BACKGROUND

Since the introduction of automatic plotting equipment in a computing environment, there has been considerable interest in producing animated films from computer programs. Some of the earliest work was done by Ken Knowlton and his group at Bell Labs, references 1,2,3,4. This Bell Labs work emphasized the use of computer-produced animated movies for the explanation of difficult concepts such as simulation of control systems, simulation of force, mass and motion interaction and other physics concepts. Among the workers

with Ken Knowlton was A. Michael Noll, whose work expressed considerable interest in using animated computer-generated graphics to illustrate concepts which are difficult to visualize without the additional coordinate of motion, references 5,6 & 7. Since the early Bell Labs work, a number of researchers have worked on systems development for simplifying computer-generated animation for the end user. Notable among these researchers were Anderson, references 8 & 9, who developed a language for aiding the user in describing the parameters of his particular animation in an easy-to-use manner, Phillips, reference 10, who reported on an animation system which made use of a low-cost remote storage tube terminal and others, references 11 & 12, whose general work was aimed at automating the animation processes in the computer. Most of this work has been concerned with producing animations for demonstrating physical phenomena in the educational environment. At the present time, many engineering groups have access to computer-controlled plotting equipment which allows them to generate and use animations in design and analysis procedures. Further discussion of some of these uses will occur in later sections of this paper.

### WHY ANIMATION?

The adage, that "one picture is worth a thousand words", can be extended to "a time sequence of pictures is worth much more than a single picture." Many times, the ability to transmit a concept or feature is enhanced by generating a time sequence of views of the phenomenon and displaying it in that fashion. In addition, as described by Noll, there are some things which are just impossible to visualize in the three-dimensional world, particularly those items which contain more than three dimensions. Consequently, if one can make use of time as representative of the additional variables, animation can be used to illustrate the salient features of the additional variables. In the particular situation of highway alignment design, it becomes very important to be able to view the proposed design of a dynamic situation so that the human response can be evaluated and used in modifying the design parameters. In this manner safety and usability are enhanced. In the case of vibratory response of items, such as structures, the response phenomena quite often is easier to see if the vibration is animated rather than if mode shapes only are plotted. The visualization of mode shapes for vibration problems is generally enhanced by a dynamic simulation as opposed to looking at only static mode plots. The rapid assimilation of large volumes of data can be made through the use of animation. Quite often animation will allow you to pick out the critical data and get right to the heart of the problem.

### TYPES OF DISPLAY DEVICES

There are a number of different types of animation which can be produced by computer techniques. The two broadest categories are animations using line or vector drawings and animations using raster scan representations. In the case of line drawings, the animation generally is generated on a Cathode Ray Tube, (CRT), device which has a grid of addressable points on the screen surface plus some vector capability for drawing lines between points on the screen. Complex views can be generated and displayed on the CRT, and through a variety of techniques, animations can be generated. In the case of the raster scan type

graphics device, scan lines are used for displaying the information on a CRT. The displays produced are similar in appearance to that of a standard television set, although for high quality work, the number of scan lines generally exceeds that of a typical television screen. Both the line type device and the raster scan type device can generate color displays. This ability to provide color can often be an important consideration in the computer-generated animation. Raster scan graphics has been used with color to do very realistic simulations of environments under computer control and some examples of this type simulation can be seen in the presentations of Greenberg and Evans.

## TECHNIQUES FOR COMPUTER PRODUCTION

There are four basic ways in which computer-generated animation can be obtained. These are

1. Through the use of photography of sequential hard copy,
2. Through the use of photography of sequential views displayed on a CRT,
3. From the direct writing on microfilm,
4. Through direct animation on the CRT.

Some of the earliest work on computer-generated animation made use of drawing sequential views with a line type plotter and then photographing these views sequentially to produce animation, (Reference 13). This technique is, of course, quite expensive since the cost to produce hard copy is high and the amount of human intervention and labor involved in performing the animation is significant. This technique has all but disappeared from use today. The second technique for producing animation is by use of direct photography of a picture generated on a CRT device. There are a number of ways of performing this operation and perhaps the most successful one is that used in many of the microfilm recorders. In this case the display buffer is dumped a single time to the CRT, generating a moving dot on the screen of the CRT with the camera lens open. When the display buffer is completely empty, the film is advanced to the next frame and you are ready to produce the next picture. This technique has often been used successfully to produce raster scan type photograph by having the raster scan only a single time with the shutter open. Some very successful work along this line was also done by Phillips at the University of Michigan (see reference 10) in which he used a storage tube type CRT and an ordinary 16-millimeter camera on which he photographed the generated display. There have also been many attempts to photograph a refresh type CRT to produce animation. There are problems associated with this method created by the refresh nature of the display CRT. In general, CRT's used for direct viewing, have a short life phosphor which is made visible by bombardment from the electron beam. Continuous visibility is obtained by periodically refreshing the display by dumping the display code from buffered memory through the electron-beam gun. Most direct-view CRT's are refreshed at from 30 to 60 times per second and, in general, the refresh rate is variable depending upon the number of line segments displayed. This variation of refresh rate creates problems of exposure control when attempting

## HIGHWAY DESIGN ANIMATION

The one area in which I have had the most experience with producing animation of actual designs has been the attempt to visually simulate a proposed highway design. In the development of a highway design, many diverse parameter effects must be evaluated as they interrelate to one another. Generally, the placement of a highway route is dependent upon study of economists, political scientists, traffic engineers, and planning and location engineers. Therefore, it is increasingly necessary to evaluate more thoroughly the interrelationship between physical design parameters and ecological and environmental consequences. Evaluation by the driver and the designer of various design alignments is also necessary to obtain the best response to all these problems.

Being able to drive a proposed highway allows the eliminating of blind spots, dangerous hilltop curves and confusing intersections and exits ramps. Better signing is also made possible. I will show some sixteen millimeter motion picture films which were generated from existing highway design data basis and made over the period of time in which the present highway animating system was developed.

In September 1968, the Washington Federal Highway Administration Office of Research and Development proposed a 18-month joint effort with the Federal Highway Administration Denver Office to produce software for the production of perspective views of proposed highway designs. Plans were made to make use of the computer hardware capability at the University of Colorado. In particular, use of the graphics capability including an interactive graphics console and a 35 millimeter microfilm recorder was planned. In September 1969, the first result of the work was shown at the 11th Annual Highway Engineering Exchange Program (HEEP) meeting in Washington, D.C. A three-minute sixteen-millimeter animation film of an actual project under design in the Denver office was shown. This film was produced from a birds-eye location of 15.24 m (50 ft) above the ground. This was done since the removal of hidden portions of the views had not yet been accomplished. In March 1970, some success was obtained in implementing an algorithm for removal of the hidden portion of the views and an animation was produced from a driver's-eye location. This development is represented by the second portion of the film which also includes a portion of actual photography of the completed highway. Both of these previously described animations were produced by photographing single frames of a refresh graphics display device and taking a number of single frames so as to produce the animation effect. As our work progressed on the perspective animation system, we developed software for producing animations directly on the 35-millimeter microfilm recorder. Consequently, the third piece of animation on the film is one of a State of California Highway Department design of a proposed road in the Six Rivers National Forest in northern California and was produced by generating single frames of equally spaced views on the 35-millimeter microfilm. The 35-millimeter microfilm was then reduced to 16-millimeter film through the use of an optical printer. This resulted in a master copy on 16-millimeter film which could then be used in the optical printing process to expose multiple frames of each of the original frames and to add color and overlays with titles and credits. It should be emphasized that all of the titles and credits were also produced on the same microfilm recorder. In 1972, the U.S. Bureau of Reclamation was involved in the design of a new water carrying canal in California and requested

to photograph such a device directly, leading to variations in intensity due to different portions of the film seeing a different number of refreshes. Similar type problems can occur in the case of raster scan devices. Additional problems may occur in the case of color devices in obtaining color saturation. Consequently, most of the direct photography work with CRT's is done with an open shutter and a single dumping of the display buffer for a single view. If the view is to be displayed in color then each color is dumped sequentially with the proper filter so as to generate maximum color saturation.

The direct writing microfilm device works by having a deflected light beam impinge directly on a sensitized film. The result is essentially identical to the photography type microfilming technique.

The last type of animation technique mentioned is that of direct animation on the CRT and perhaps this is the one which many of us have been aiming towards from the beginning. Unfortunately, in many engineering design problems, the amount of computation involved in producing a animated sequence is still too much to allow for real-time display of the results. As the animating portions of CRT display devices are becoming implemented more often in hardware, this problem appears to be lessening. At the present time, one can find hardware implementations for rotations, scaling, perspective transformations, intensity queuing, and in some cases, hidden line removal. All of these implementations have made it easier to get to a real-time animated display directly on the CRT device.

## APPLICATIONS

Some of the earliest proposed animation systems were used to draw stick like cartoons or for the production of animated art work. More recently, there has been considerable interest in using raster scan color graphics to produce animated television commercials. A primary interest of this group is the use of computer-generated graphical animations in the area of simulations. There is a wide range of applications in this simulation area, ranging from simulating the response and behavior of a structure or engineering design under some time varying situation all the way to attempting to simulate the real surrounding areas as is the case for pilot and driver training simulations. The examples of work which I will show with the film clips are really concerned with attempting to combine the two types of simulation - that of superposition of an engineering design on the surrounding real world and an evaluation of this simulation both from an engineering design and an aesthetic point of view.

High quality computer-generated animations have been produced at a wide variety of installations, including the University of Utah, Cornell University, General Electric Syracuse, National Center for Atmospheric Research, Sandia Labs, and Livermore, just to mention a few.



that we produce an animation of this project so that they could evaluate the effects of the proposed design on the surrounding terrain. The fourth portion of the film which you will see contains this animation of flying over the proposed canal. This piece of animation was produced with the identical technique used for the US 199 animation in California with the exception that some of the colors added with the optical printing process are different.

The fifth and final film portion is a copy of an animated sequence of a highway design produced for the State of New York Department of Transportation by General Electric Syracuse on their raster scan color graphics system.

## CONCLUSIONS

The use of computer-generated animations to aid in engineering analysis and design appears to have a bright future. The developments of faster computational hardware, along with the incorporation of many graphical display features into hardware point to increasing ability to provide real-time animations directly on CRT screens. Meanwhile, many important engineering problem solutions will be aided by using microfilm recorder techniques to produce animations.

## REFERENCES

1. Knowlton, K.C., "A Computer Technique for Producing Animated Movies," AFIPS Conference Proceedings, Vol. 25, P. 64, 1964.
2. Knowlton, K.C., "Computer Generated Movies, Design and Diagrams," Design and Planning -2, p. 59, Hastings House, N.Y., 1967
3. Zajac, E.C., "Simulation of a Two Gyro, Gravity Gradient Attitude Control System," Bell Telephone Laboratories (16mm film, 4 min., sound).
4. Sinder, F.W., "Force, Mass and Motion," Bell Telephone Laboratories (16mm film, 10 min., sound).
5. Noll, A.M., "A Subjective Comparison of Piet Mondraings' Compositions with Lines and a Computer Generated Picture," Psychological Record, Vol. 16, p. 1, Jan. 1966.
6. Noll, A.M., "A Computer Technique for Displaying n-Dimensional Hyperobjects," Communications of ACM, Vol. 10, No. 8, Aug. 1967.
7. Noll, A.M., "Computer Animation and the Fourth Dimension," AFIPS FJCC Proceedings, p. 1279, 1968.
8. Anderson, S.E. and Weiner, D., "A Computer Animation Movie Language for Educational Motion Pictures," AFIPS FJCC Proceedings, p. 1317-1320, 1968.
9. Anderson, S.E., "Generating Computer Animated Movies from a Graphic Console," Advanced Computer Graphics, Edited by R.D. Parslow and R.E. Gree, Plenum Press, pp. 717-722, 1971.

10. Phillips, R.L., "Production of Computer Animated Films from a Remote Storage Tube Terminal," Advanced Computer Graphics, Edited by R.D. Parslow and R.E. Green, Plenum Press, pp. 723-731, 1971.
11. England G.A., et al. "Automating Animation," Online 72 Conference Proceedings, Vol. 2, p. 445, Sept. 1972.
12. Noland, J.F. and Yarbrough, F.D., " An On-Line Computer Drawing and Animation System," Proceedings, IFIPS Congress '68, North Holland Publishing Co, Amsterdam, p. C103, 1968.
13. Godin, P., et al., "Visual Quality Studies in Highway Design," Highway Research Record No. 232, pp. 46-57, 1968.

**Page Intentionally Left Blank**

THE CONTROL DATA "GIRAFFE"\* SYSTEM FOR  
INTERACTIVE GRAPHIC FINITE ELEMENT ANALYSIS

S. Park and D. M. Brandon, Jr.

Control Data Corporation

SUMMARY

This paper describes the Control Data General Purpose Interactive Graphics Application Package "GIRAFFE" (Graphical Interface for Finite Elements). The system is a general purpose interactive graphics pre/post processor for structural analysis computer programs. The system will facilitate the engineer to create, edit, or review all the structural input/output data on a graphics terminal in a time-sharing mode of operation.

INTRODUCTION

Structural Analysis problems encountered in engineering design tend to be inherently complex in nature. Establishing the structural idealization which approximates the true structure constitutes a prime problem. Also the visualization and interpretation of the solution results from the computer program is a major problem.

Traditionally, the finite element idealization is created by the engineer using structural drawings. This procedure requires a significant amount of the engineering effort due to the typical requirements of the structural computer program (element nodal coordinates, element connectivity information, etc.). The input information is usually generated as rigid format computer cards.

Depending on the complexity of the structure, this procedure results in approximately 70% of the engineer's time spent on the input data preparation, 20% for the solution interpretation and only 10% for the actual computer runs.

The General Purpose Interactive System "GIRAFFE" is developed in such a manner that the engineer can create, edit or review all the structural input/output data on the graphics terminal in a time-sharing mode of operation. Thus, GIRAFFE will enable the finite element programs to truly become more viable tools for large complex analysis problems in the structural engineering community.

Overall objectives of the GIRAFFE software are shown in Figure 1. The functional capabilities available in GIRAFFE Version 1 and Version 2 are shown in Figure 2 and Figure 3.

---

\* This does not represent any official intent at this time to use the development name "GIRAFFE" as a product name.

## OPERATING ENVIRONMENT

The operating environment of the GIRAFFE system is shown in Figure 4. The interactive graphic pre and post processing takes place in a CDC CYBER/6000 using the SCOPE/INTERCOM operating system. The terminals initially supported are Tektronix 4000 series (such as the 4010 or preferably the larger screen 4014) and the CDC CYBER GRAPHICS (777).

GIRAFFE exclusively uses the CDC GODAS (Graphically Oriented Design and Analysis System) graphic utility software. This provides for the most comprehensive graphics capabilities available such as:

- . Free format - total error recoverable dialog and graphic I/O.
- . Three-dimension interactive display creation and editing.
- . Virtual address data base and vector data base manager capability.
- . Automatic screen control (split screen usage on DVST<sup>\*</sup> terminals) with local (application independent) interrupt system for 2-D zooming, 3-D transformations, "hard copy cleaning", off-line plot file accumulation, etc.

The CDC GODAS uses the CDC IGS (Interactive Graphics System/basic graphic software) package for the appropriate terminal. This allows the terminals to be driven at synchronous communication rates (200 - 9600 BAUD) for faster user response.

The Finite Element Solution Phase may be run on a variety of machines using the chosen structure's application programs. The GIRAFFE system uses a "Neutral Element Library" (for finite element type definition) and produces/processes a "Neutral I/O<sup>+</sup> file" for input to and output from the structure's programs.

GIRAFFE also provides for alternative sources of surface definition, in addition to direct generation, pictorially or key-in on the Graphics terminal. These alternate sources may originate from digitizers, surface scanners or interactive drafting systems.

Surface macro, which is a function of the surface definition task, provides to the user very powerful capabilities such as rotation, translation and mirror image of a basic surface. Thus, a detailed geometrically symmetric or similar structure can be easily created from a simple basic surface defined initially.

---

\* Direct View Storage Tube

+ See the section entitled Neutral I/O Files

The data created in the Structural Data Base (SDB) by GIRAFFE can be saved permanently for the future usage. For example, if a portion of the structural component is changed at a later date, the saved tape of that SDB can be brought in to change that affected portion through surface modification or mesh modification, then perform structural analysis without recreating the whole structural data.

Under the design philosophy employing structural data base, GIRAFFE provides practically unlimited problem size capability (for example up to 30,000 elements and 35,000 nodes).

#### NEUTRAL ELEMENT LIBRARY

GIRAFFE provides a set of Neutral Structural Elements. The user of GIRAFFE will select one or a multiple of Neutral Elements to suit his finite element modeling requirements.

Translation from the user selected Neutral Element to the corresponding finite element of the specific application program which performs the solution phase will be handled within the application program. This gives complete freedom to the user to select the best suited program for the type of structural analysis he is performing.

The Neutral Elements available in the Version 1 of GIRAFFE are listed in the following table:

<u>Element Name</u>	<u>Description</u>
EN2TRUS .....	Truss element
EN2BEAM .....	Beam element
EN2TUBE .....	Tube element
EN2LSP .....	Linear translational spring element
EN3TSP .....	Linear torsional spring element
EN3MEM .....	Triangular membrane element
EN3PLT .....	Triangular plate element
EN3PM .....	Triangular plate element with membrane, bending and shear in effect
EN3PM1 .....	Triangular composite element
EN4MEM .....	Quadrilateral membrane element
EN4PLT .....	Quadrilateral plate element
EN4PM .....	Quadrilateral plate element, with membrane, bending and transverse shear in effect
EN4PM1 .....	Quadrilateral composite element

#### NEUTRAL I/O FILES

The Neutral I/O files are generated automatically. When the user completes input data preparation using GIRAFFE, he generates a "Neutral Input" file through GIRAFFE. He will subsequently submit a batch job to execute the structural application program (whichever he chooses) with the Neutral Input file as input source.

When Neutral Output file returns to a Graphics Computer it is attached as the Neutral Output file accessible to GIRAFFE. The Neutral Output file contains displacements and stresses. GIRAFFE V.1 will provide post processing activities such as deformation and stress analysis. GIRAFFE V.2 will provide deformation, stress plots and modal shape plots, and time response history plots.

#### HIGHLIGHTS OF THE GIRAFFE MODULES

GIRAFFE is an integrated pre/post processor to a structural application program with its own data base (see figure 5). In an interactive graphic time-sharing mode, GIRAFFE facilitates generation of the structures element modeling; i.e., element connectivities and node points data.

The objective of the GIRAFFE does not end here but is extended further. It allows preparation of all the input to a structural application program. The complete task options available in Version 1 are shown in the Master Menu listed below as displayed on terminal.

##### Task Options:

- 1 Terminate
- 2 Surface definition
- 3 Element mesh generation
- 4 Element mesh geometry check
- 5 Structure's property definition
- 6 Restraint definition
- 7 External load definition
- 8 Bandwidth optimization
- 9 Write neutral input file
- 10 Read neutral output file
- 11 Deformation analysis
- 12 Stress analysis
- 13 Status

Enter Task Number

The rest of this section shows only the main menu list for each task in the GIRAFFE V.1.

#### (1) SURFACE DEFINITION

This is the basic surface (outline or part definition) task. This task allows the user to create a basic surface (whether the source is direct CRT or batch input) and modify or delete surface data from the SDB.

##### Surface Definition Options

- 1 Return
- 2 CRT input
- 3 Batch input/modify
- 4 Surface macro

Enter task number

## (2) ELEMENT MESH GENERATION

This task generates a finite element model either completely automatically or one-by-one manually. Also, elements can be modified. Finite element types are defined in this phase:

Element Mesh Generation:

- 1 Return
- 2 Select other structure and/or block
- 3 Define sub-block for mesh generation
- 4 Auto mesh
- 5 Manual mesh - define/edit
- 6 Delete mesh generated

Enter Task Number

## (3) ELEMENT MESH GEOMETRY CHECK

Under this option the user can perform the following element mesh geometry check:

Element Mesh Geometry Check:

- 1 Return
- 2 Select other structure and/or block
- 3 Show block mesh boundary
- 4 Quadrilateral planarity check
- 5 Element normal check
- 6 Define or modify element normal

Enter Task Number

## (4) STRUCTURE'S PROPERTY DEFINITION

Under this option the user can define a structure's material properties and physical properties; also, output stress reference angle. The user can request material and physical property ID numbers to be displayed on the element mesh.

Property Definition:

- 1 Return
- 2 Select other structure/block
- 3 Define material properties
- 4 Define physical properties
- 5 Display/cancel mode numbers
- 6 Display/cancel element numbers
- 7 Display/cancel material property ID
- 8 Display/cancel physical property ID

Enter Task Number



## (5) RESTRAINT DEFINITION

Under user's direction, this task restrains freedoms of the nodes. The user can also define enforced boundary node displacements of the nodes. The main task menu is listed below as displayed on terminal.

Restraint Generation:

- 1 Return
- 2 Other structure/block
- 3 Define restraints
- 4 Display restrained nodes
- 5 Define boundary displacements
- 6 Display displaced nodes
- 7 Display/cancel node number
- 8 Display/cancel element numbers

Enter Task Number

## (6) EXTERNAL LOAD DEFINITION

This task assigns external loads to the nodes or elements upon user's commands. GIRAFFE V.1 load options are:

- . Concentrated load(s)
- . Pressure load(s)
- . Gravitation load
- . Beam load(s)

Multiple load case definition is allowed. The main menu for this task is listed below:

Load Options:

- 1 Return
- 2 Other structure/block
- 3 Load input
- 4 Display load
- 5 Display/cancel node numbers
- 6 Display/cancel element numbers
- 7 Change Load Case

Enter Task Number

## (7) BANDWIDTH OPTIMIZATION

This task selection allows the user to resequence node numbers to reduce matrix (stiffness) bandwidth for solution efficiency. This is an optional task. The user who wants to utilize built-in application bandwidth optimizer may not exercise this option. The main menu for this task is:

Bandwidth Optimization:

- 1 Return
- 2 Select other structure

- 3 Optimize bandwidth
- 4 Display/cancel node number

Enter Task Number

#### (8) WRITE NEUTRAL INPUT FILE

This is the last task of the GIRAFFE pre-processing phase. When the user completes the generation of all the necessary input data for structural analysis, he will select this option to generate neutral input file. This file will contain all the necessary and compatible data for the structural application programs. Upon exit from GIRAFFE he submits a batch application job with the neutral input file as the input source.

GIRAFFE dialogs in this task are:

- . Which structure's neutral input file is to be generated if there are multiple structures in SDB.
- . Build load set from the predefined load case in the load module.

#### (9) READ NEUTRAL OUTPUT FILE

This task reads the application program solution data saved in neutral output file. The user can subsequently process stress and displacement data in the interactive graphics mode.

GIRAFFE dialogs in this task include:

- . Enter neutral output file set number corresponding to previously created neutral input file set.
- . Results analysis for a single load set.
- . Results analysis via load set superposition.

#### (10) DEFORMATION ANALYSIS

Under this task the user can generate deformation plots superimposed with original structure with boundary lines only, or he can create entire mesh deformation superimposed with original mesh. The latter option should be used for detail deformation study associated with a specific block. The main menu of this module is shown below.

Deformation Analysis:

- 1 Return
- 2 Other block(s)
- 3 Limit deflection node display
- 4 Boundary deformation plot
- 5 Entire element deformation plot
- 6 Display/cancel node numbers

Enter task number

## (11) STRESS ANALYSIS

Under this task control, the user can investigate stress distribution of the structure for the applied load condition. The main menu of this module is shown below.

Stress Analysis:

- 1 Return
- 2 Other block(s)
- 3 Beam-type element stress
- 4 Element stress display
- 5 Stress contour plot
- 6 Display/cancel node number(s)
- 7 Display/cancel element number(s)

Enter Task Number

## (12) STATUS

This task provides the user some essential status summary of the GIRRAFE usage plus structural data base interpretive dump.

Status Summary:

Time: \_\_\_\_\_  
Elapsed clock time this session: \_\_\_\_\_  
Total elapsed clock time: \_\_\_\_\_  
CPU time this session: \_\_\_\_\_  
Total elapsed CPU time: \_\_\_\_\_  
Page size: \_\_\_\_\_  
Number of page transfers<sup>+</sup> this session: \_\_\_\_\_  
Total number of page transfers: \_\_\_\_\_  
Amount of disk space allocated: \_\_\_\_\_  
Number of structure: \_\_\_\_\_  
    Structure: \_\_\_\_\_  
    Name: \_\_\_\_\_  
    Number of blocks: \_\_\_\_\_  
    .  
    .

---

<sup>+</sup> This is transfer between central memory and disk

The interpretive SDB dump options are listed below:

Enter Option Number:

- 1 Return
- 2 Structure list
- 3 Block list
- 4 Surface line lists
- 5 Surface polygon list
- 6 Mesh list
- 7 Boundary line list
- 8 Elements list
- 9 Node point list
- 10 User node number and resequenced node number
- 11 Material and stress angle list
- 12 Physical property name and value list
- 13 Material property name and value list
- 14 Load lists
- 15 Boundary displacement lists
- 16 Node displacements
- 17 Output reaction force lists
- 18 Stress list
- 19 Element library list

#### STRUCTURAL DATA BASE

The Structural Data Base can be regarded as an unlimited memory where data is organized in lists. The main purposes of the SDB are to provide a permanent record of Structural Data and to provide communication between functional modules and for system save/restart.

The data base manager system allows the user to create, access, and modify lists on a random basis. The main purpose of the SDB Manager is to free the functional module from all I/O concern and associated code.

The data base requirements of three-dimensional analytic geometry software and other structural tasks are quite complex:

- 1 Large amounts of data must be manipulated.
- 2 A variety of algorithms and procedures are used which require access to data in both a sequential and random fashion.
- 3 A variety of different data "collections" are involved.
- 4 Maximum efficiency is mandatory to provide tolerable response time to the user at the terminal.

The GIRAFFE data base is composed of two files: one for real numbers and the other typically for integer numbers. Each of these two files logically consists of a list of sequential binary data words. The two files

physically consist of a collection of "pages" (Data Records). Data collections or "subfiles" are indicated by starting address pointers, as shown in Figure 6. The data base routines make the computer appear to be an efficient "virtual memory" device, handling all mapping, paging, and I/O.

A number of real pages and integer pages will be held in central memory for each of the two files. Pages are automatically "swapped" into central memory as required (or demanded). Pages are swapped out of central memory according to a "least recently used procedure".

The Data Base Manager has the additional capability to perform "macro" operations as well as the I/O of individual words. These macro routines take the overhead (of mapping, swapping, etc.) involved and reduce it by the page size (or length of the macro operation if less than the page size). The macro routines also perform the necessary arithmetic and data movement functions simultaneously with the I/O, thus considerably reducing the programming effort and storage required. Figure 7 shows some macro capabilities which are extensively used in analytical geometry and structural engineering calculations. This operation is conceptually shown in Figure 8.

The overall system is optimized (trading off central memory storage with central memory residency interval (CP & I/O time) by simply manipulating pages sizes (real and integer).

With one Data Base and Data Base Manager, development, documentation, and continuation costs are reduced and flexibility for modifications and additions is increased.

#### GIRAFFE APPLICATION EXAMPLE

Some of the major structural tasks are illustrated in Figures 9 through 25 for a simple 3D plate problem.

### OVERALL OBJECTIVES

- . An innovative, powerful, flexible design/analysis tool for the structural engineering community.
- . Cost/effective software system implementation designed for efficient use in an interactive time-sharing environment.
- . Terminal independence (i.e., structural data base can be accessed by any GODAS-supported terminals either low cost DVST or high performance CDC 777 refresh terminal or future undefined high performance devices).
- . Large reduction in user's design and redesign cycle time.
- . Minimum user capital outlay for Graphics terminal.

Figure 1.- Overall objectives.

## MAJOR FUNCTIONAL CAPABILITIES

### SYSTEM

No practical limit on problem size  
Ability to continue interrupted work  
On-line real time hard copy plot  
GERBER, CALCOMP, Houston Instrument, SC4020 plots can be on/off-line processed

### DISPLAY UTILITY

Windowing  
Zooming  
Basic transformations (rotation, etc.)  
Other transformations (perspective, etc.)  
Hidden line removal on surface definition

### PREPROCESSING

Basic surface definition  
Model geometry check; boundaries, normals and planarity of quadrilateral elements  
Node and/or element number display and delete  
Element generation and edit can be in any sequence  
Property specification  
Material property ID display  
Physical property ID display  
Automatic 2-D mesh generators  
Automatic 3-D mesh generators (plate and shell problems)  
Surface definition input (interactive)  
External forms of surface definition input  
Manual mesh generator/editor  
Bandwidth optimizer (Gibbs-Poole-Stockmeyer method)  
Load specification  
Display and edit load  
Restraint generator  
Display and edit restraint

### INTERFACE TO SOLUTION PHASE

I/O interface to various analysis packages: neutral I/O scheme  
Neutral element library (classical elements)

### POSTPROCESSING

Display of stress values including critical limit  
Display of deformations including critical limit

Figure 2.- Major functional capabilities (Version I).

## MAJOR FUNCTIONAL CAPABILITIES

### SYSTEM

No practical limit on problem size  
Ability to continue interrupted work  
Can merge data base to form a new large data base (SDB)\*  
Can extract one data base from other  
On-line real time hard copy plot (on Tektronix 4000 series)  
Gerber, CALCOMP, Houston instrument, SC4020 plots can be on/off-line processed

### DISPLAY UTILITY

Windowing  
Zooming  
Basic transformations (rotation, etc.)  
Hidden line removal  
Other transformations (perspective, etc.)

### PREPROCESSING

Basic surface definition  
Model geometry check; boundaries, normals, and planarity of quadrilateral elements  
Node and/or element number display and delete  
Element generation and edit can be in any sequence  
Property specification  
Material property ID display  
Physical property ID display  
Automatic 2-D mesh generators  
Automatic 3-D mesh generators (plate and shell problems)  
Automatic 3-D solid mesh generators  
Surface definition input (interactive)  
External forms of surface definition input  
Manual mesh generator/editor  
Bandwidth optimizer (Gibbs-Poole-Stockmeyer method)  
Curved surface mesh generation (isoparametric elements)  
Load specification  
Display and edit load  
Restraint generator  
Display and edit restraint  
Temperature specification  
Display and edit temperature model

### INTERFACE TO SOLUTION PHASE

I/O interface to various analysis packages: Neutral I/O scheme  
Neutral element library (classical elements)  
Neutral element library (isoparametric, solid and special elements)

### POSTPROCESSING

Display of stress values including critical limit  
Stress contour plot  
Display of deformations including critical limit  
Boundary deformation W/W overlapping original boundary  
Load case superposition to produce composite deflection plot  
Modal shape plot  
Structural time response history plot

\* Structural Data Base

Figure 3.- Major functional capabilities  
(Version II).



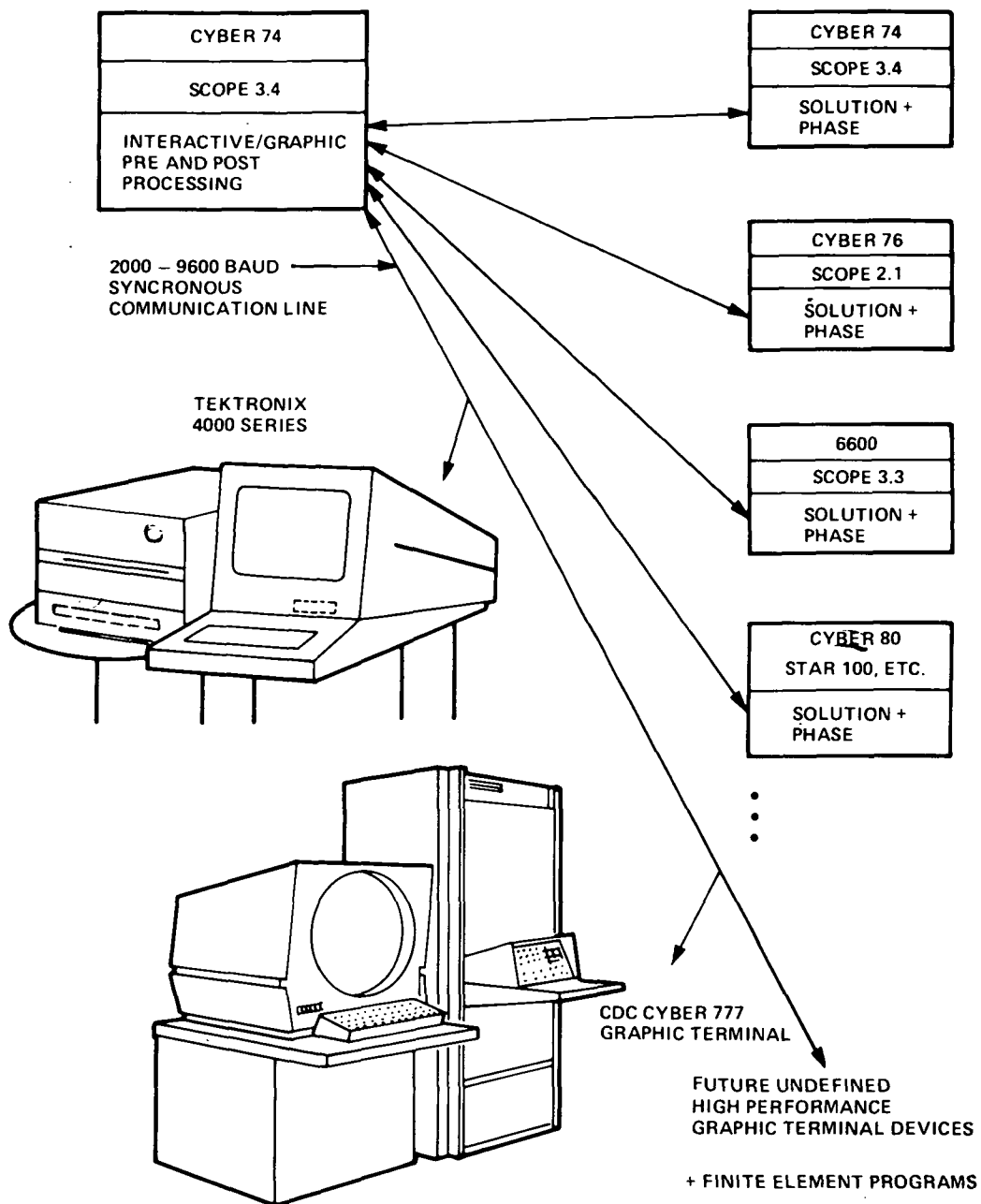


Figure 4.- Operating environment.

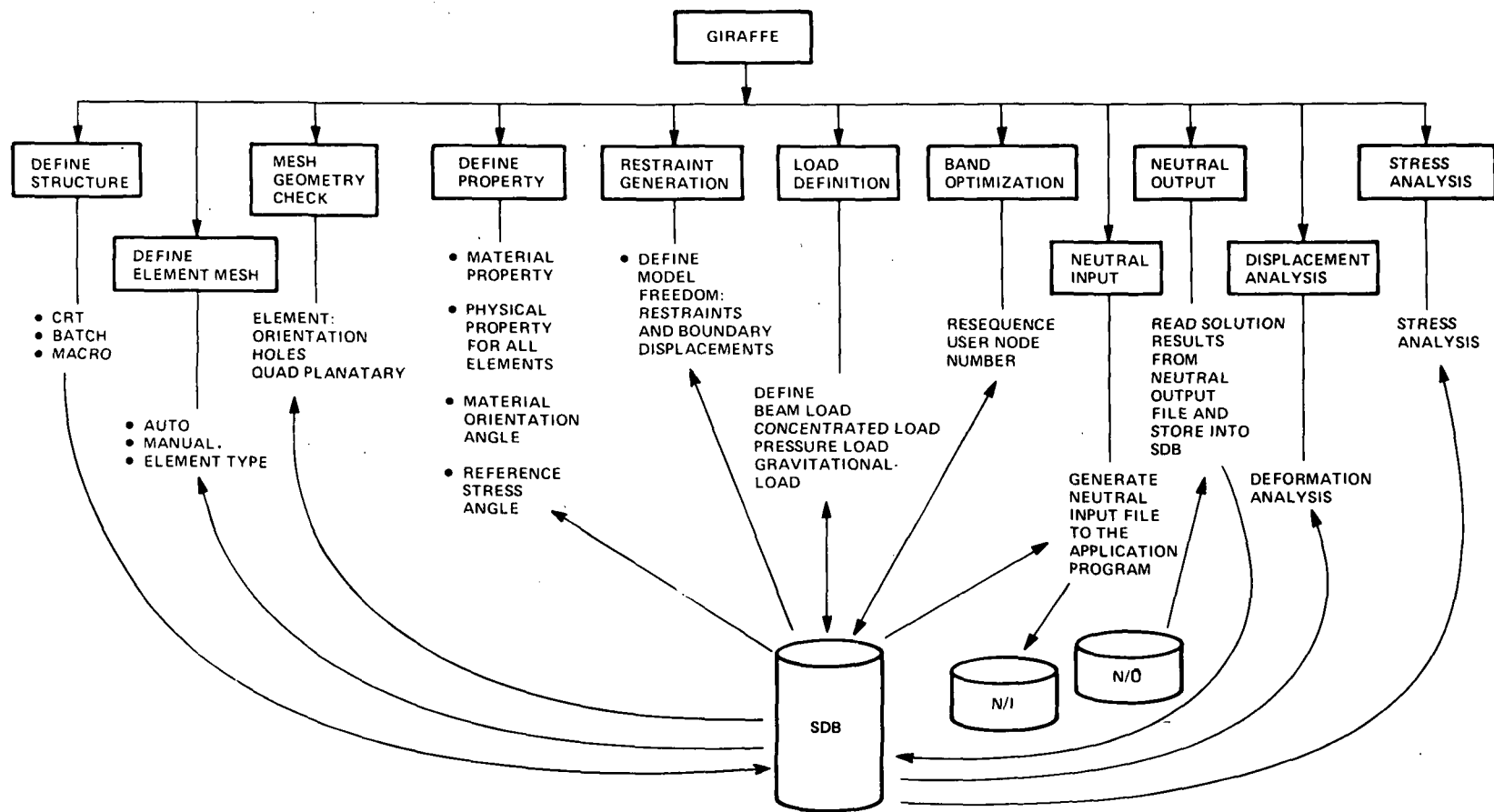


Figure 5.- GIRAFFE functional modules and their task summaries.

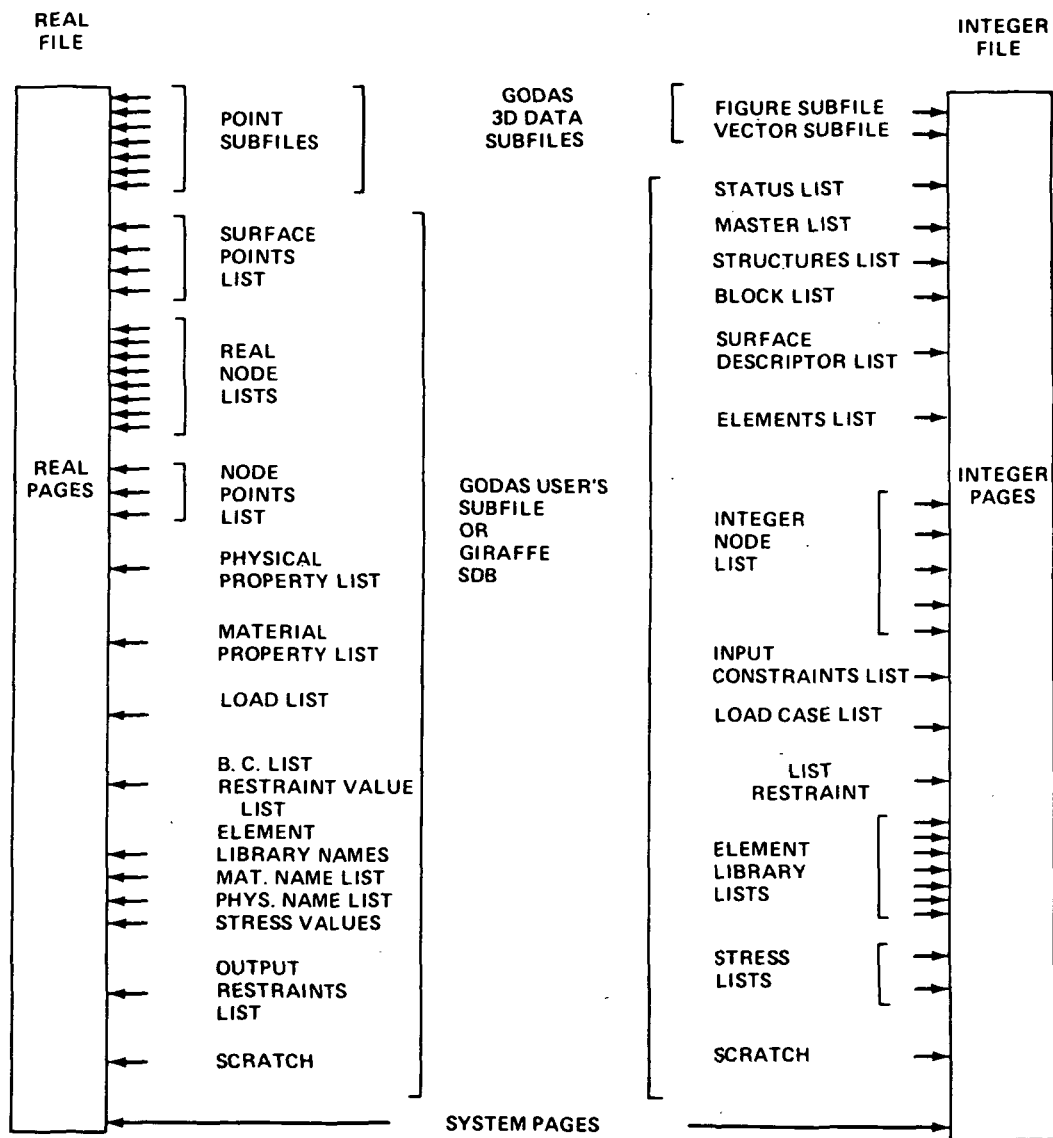


Figure 6.- Conceptual view of structural data base organization.

DATA BASE MANAGER  
MACROS AND THEIR OPTIONS\*\*

o MOVE (GDMOVER AND GDMOVI\*)

$$\begin{aligned}\bar{A} &\rightarrow \bar{B} \\ -\bar{A} &\rightarrow \bar{B} \\ \bar{A} &\rightarrow \bar{B} \quad (\text{ELEMENT BY ELEMENT ABSOLUTE VALUE}) \\ \bar{A} * C &\rightarrow \bar{B} \\ \bar{A} + C &\rightarrow \bar{B}\end{aligned}$$

o NORM (GDNORMR)

$$\begin{aligned}\Sigma \bar{A} &\rightarrow C \\ \pi \bar{A} &\rightarrow C \\ \text{MAX } \bar{A} &\rightarrow C \\ \text{MIN } \bar{A} &\rightarrow C \\ \text{MAX } \bar{A} &\rightarrow C \\ \text{MIN } \bar{A} &\rightarrow C \\ \text{MIN (POS } A) &\rightarrow C \\ \text{MAX (NEG } A) &\rightarrow C\end{aligned}$$

o VECTOR (GDVECTR)

$$\begin{aligned}\bar{A} + \bar{B} &\rightarrow \bar{A} \\ \bar{A} - \bar{B} &\rightarrow \bar{A} \\ \bar{A} * \bar{B} &\rightarrow \bar{A} \\ \bar{A} / \bar{B} &\rightarrow \bar{A} \\ \bar{A} \cdot \bar{B} &\rightarrow C \quad (\text{DOT PRODUCT}) \\ \bar{A} - C \cdot \bar{B} &\rightarrow \bar{A} \quad (\text{ROW OPERATION})\end{aligned}$$

o EXTERNAL (GDEXTR AND GDEXTI\*)

$$\begin{aligned}\bar{E} &\rightarrow \bar{A} \quad (\text{PUT ARRAY INTO DATA BASE}) \\ C &\rightarrow \bar{A} \quad (\text{BROADCAST CONSTANT}) \\ \bar{A} &\rightarrow \bar{E} \quad (\text{TAKE ARRAY OUT OF DATA BASE})\end{aligned}$$

WHERE  $\bar{E}$  IS AN EXTERNAL (USER DEFINED) ARRAY - MAY BE A PORTION OF PAGING BUFFER ALSO VIA RELEASE AND RESERVE CAPABILITIES

\* FOR INTEGER MACROS

\*\* OTHER MACROS PROVIDE FOR SPECIAL FUNCTIONS AS SPARSE MATRIX (VECTOR) OPERATIONS, MASKING, SELECTION, ETC.

Figure 7.- Data base manager macros.

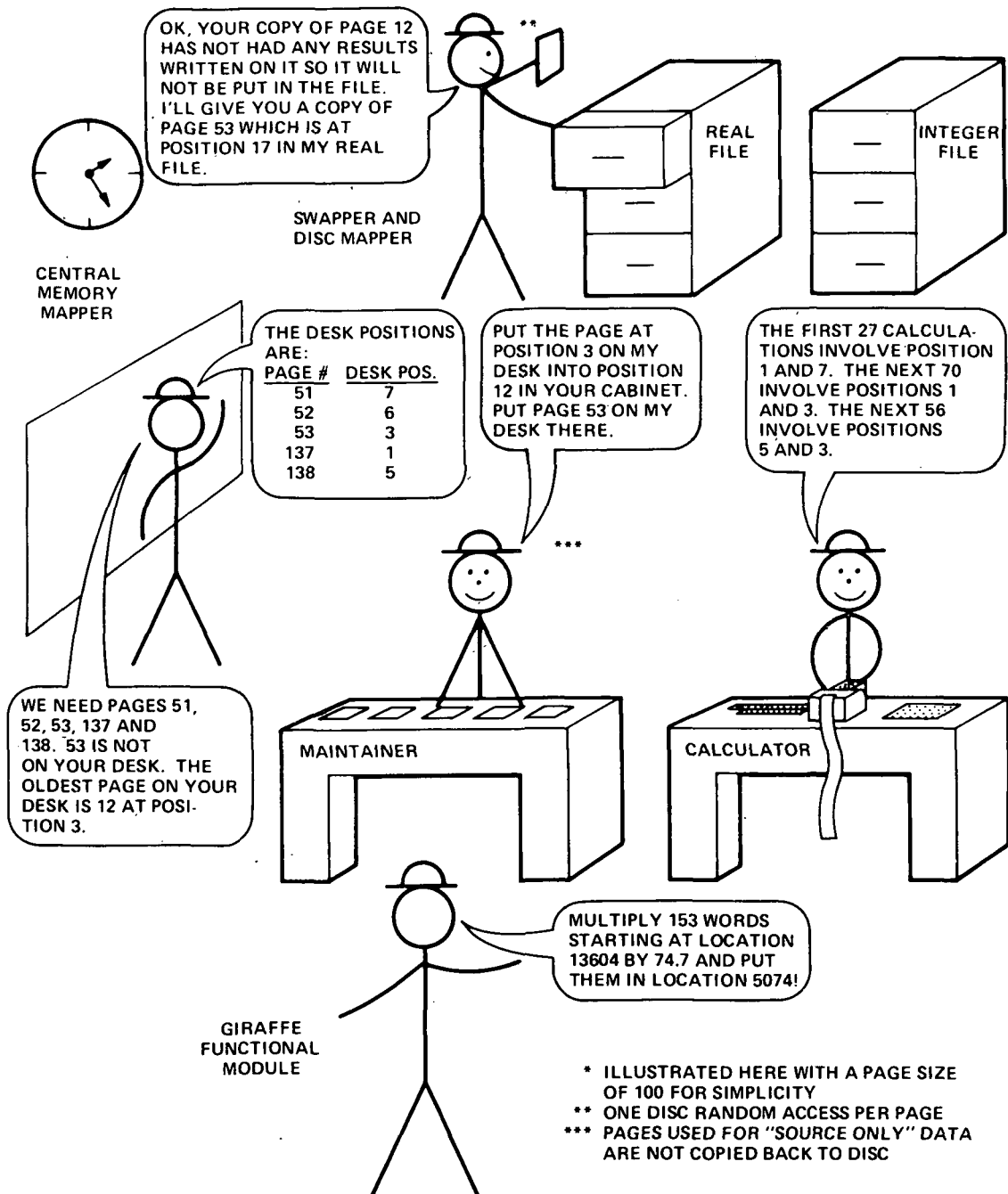


Figure 8.- Structural data base operation\*.

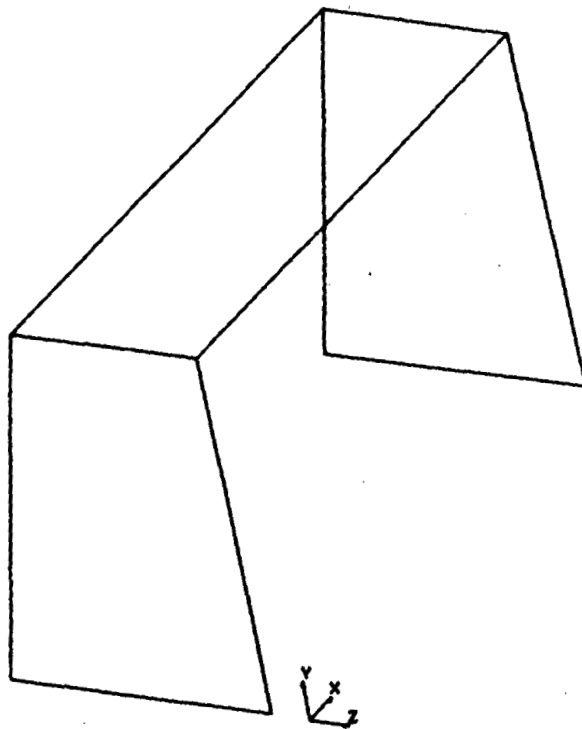


Figure 9.- Basic structure surface definition display  
(CRT, batch or other input).

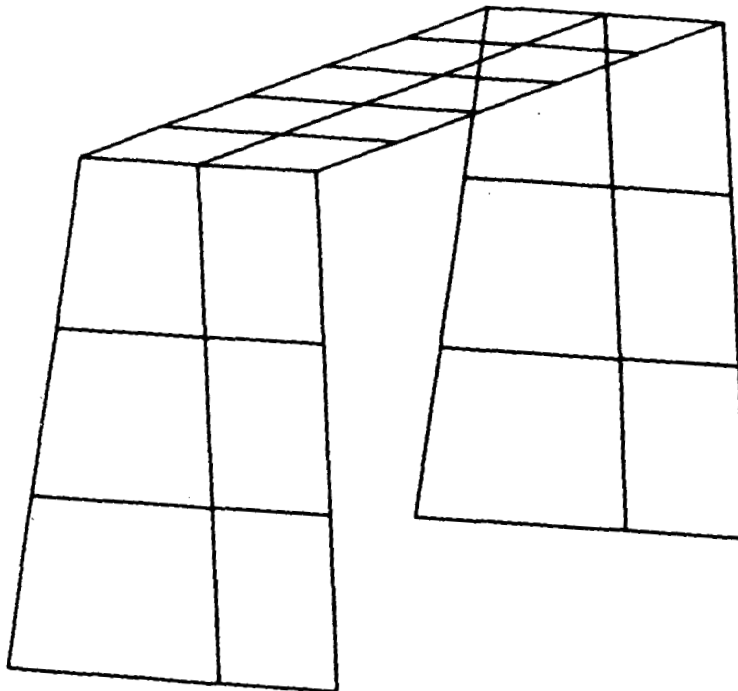


Figure 10.- Structure with finite element mesh.

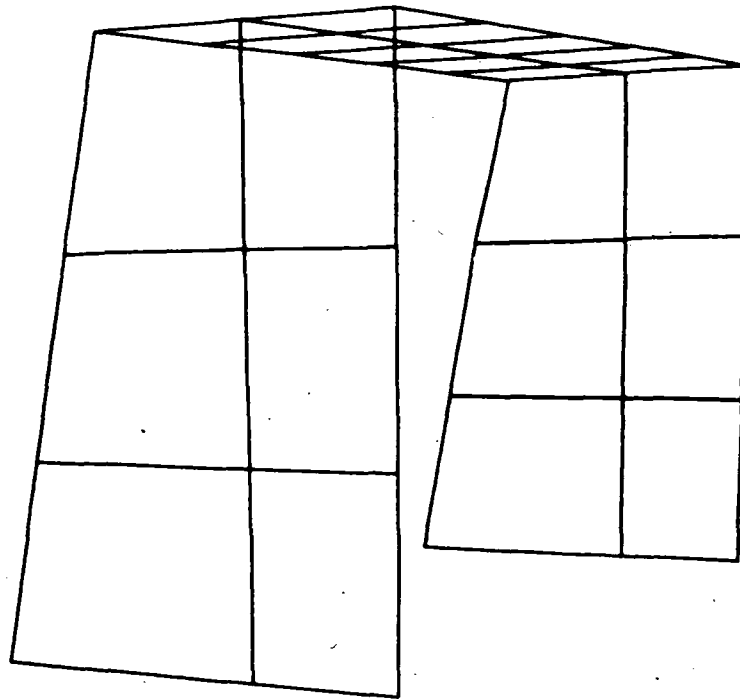


Figure 11.- Perspective view of finite element mesh.

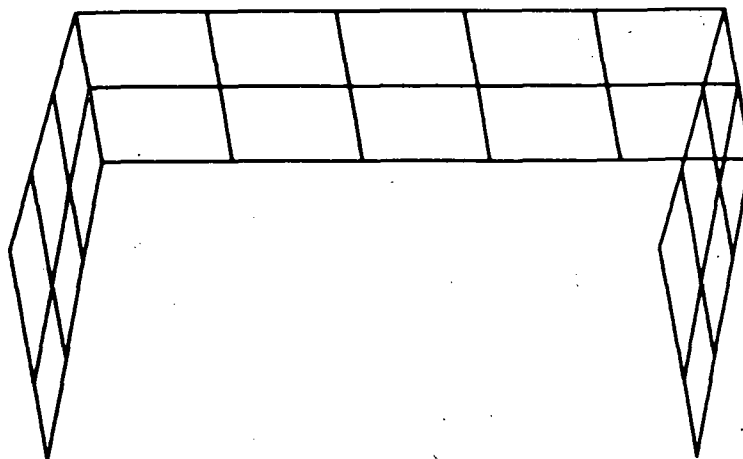


Figure 12.- Rotated view of structure.

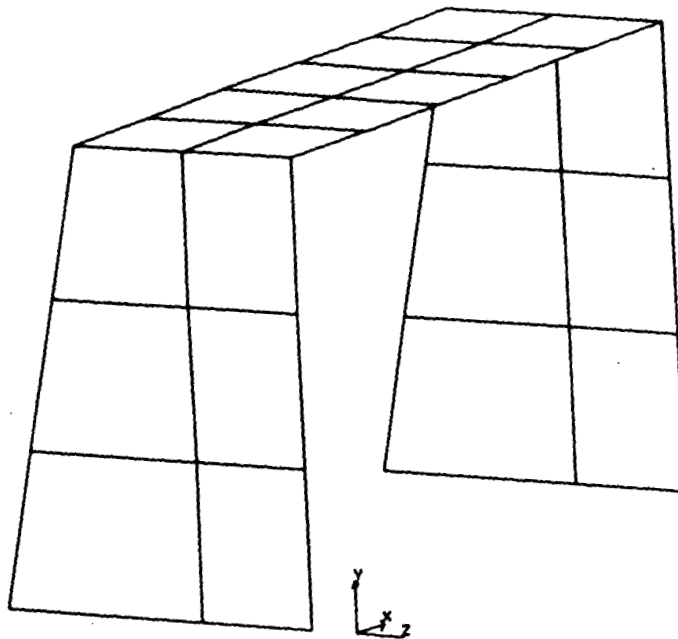


Figure 13.- Structure with hidden lines removed.

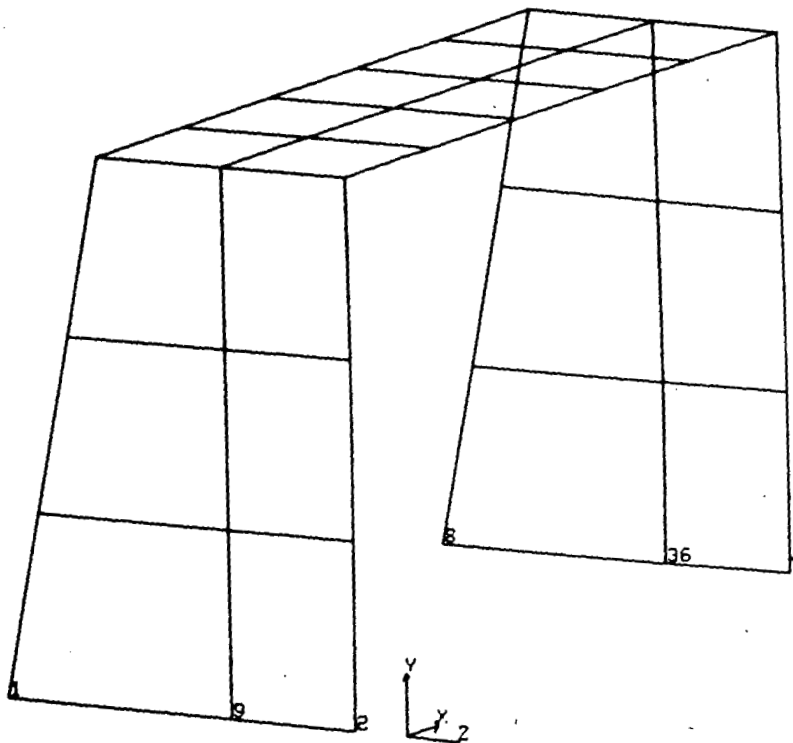


Figure 14.- Fully restrained node number display  
(restrained in all six degrees of freedom).



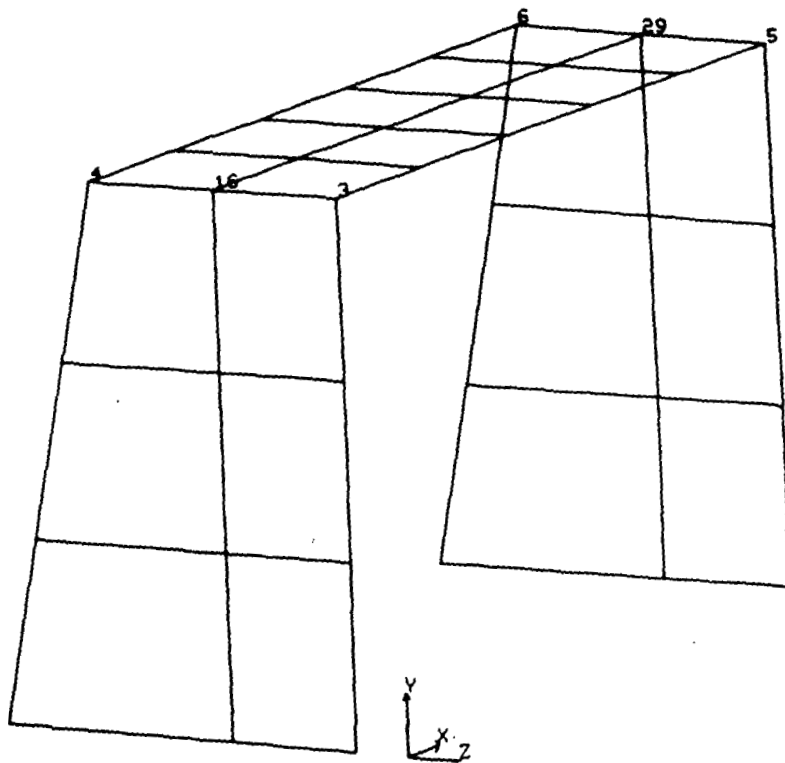


Figure 15.- Node number display for unrestrained nodes.

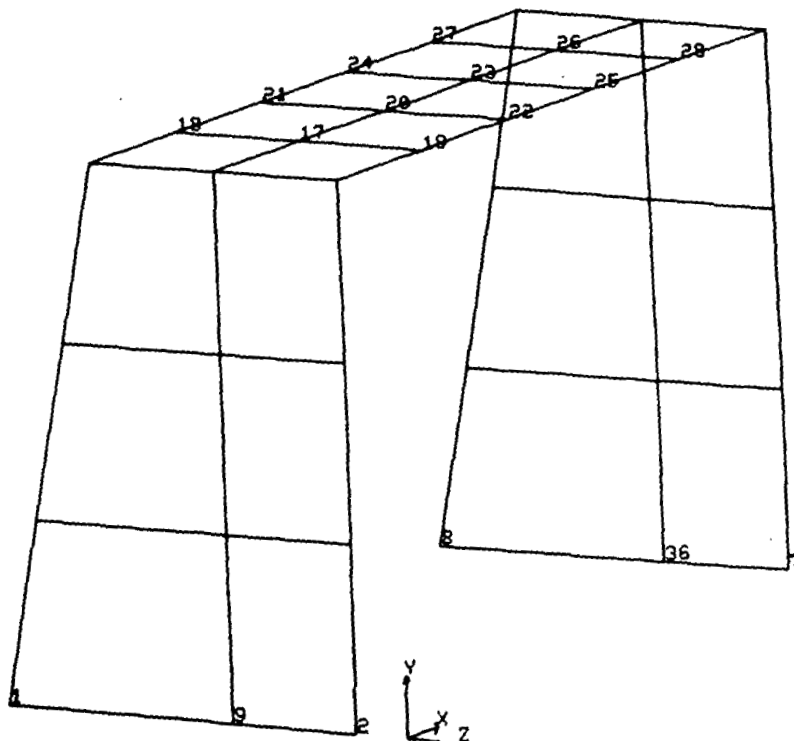


Figure 16.- Nodes restrained in RY degree of freedom.

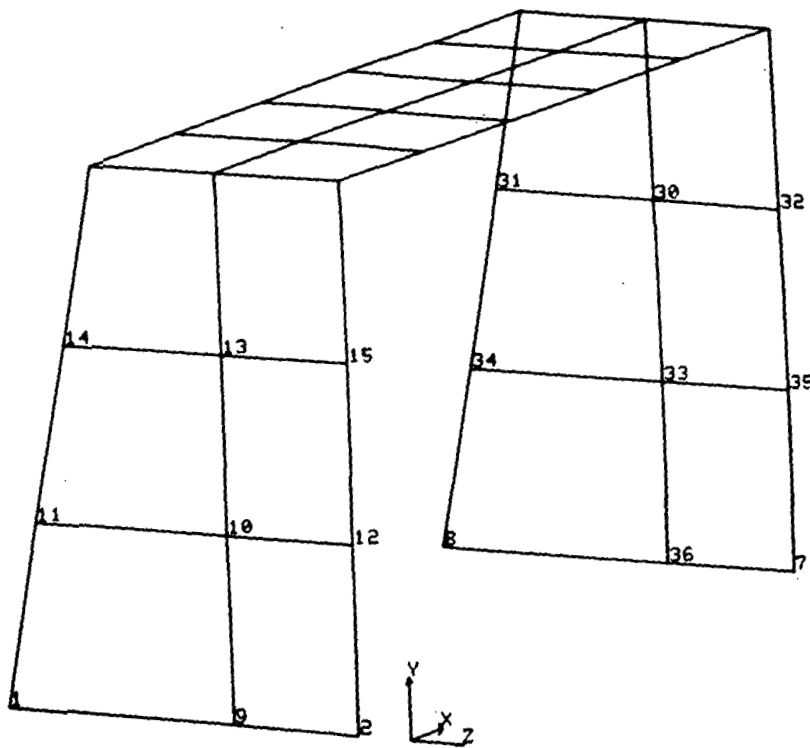


Figure 17.- Nodes restrained in RX degree of freedom.

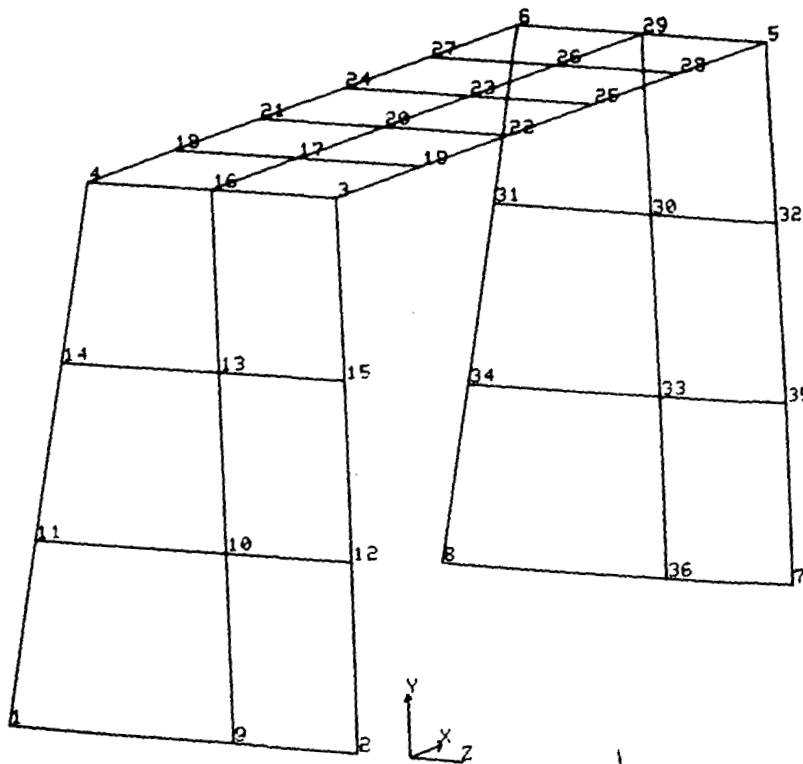


Figure 18.- Finite element model with user node numbering.

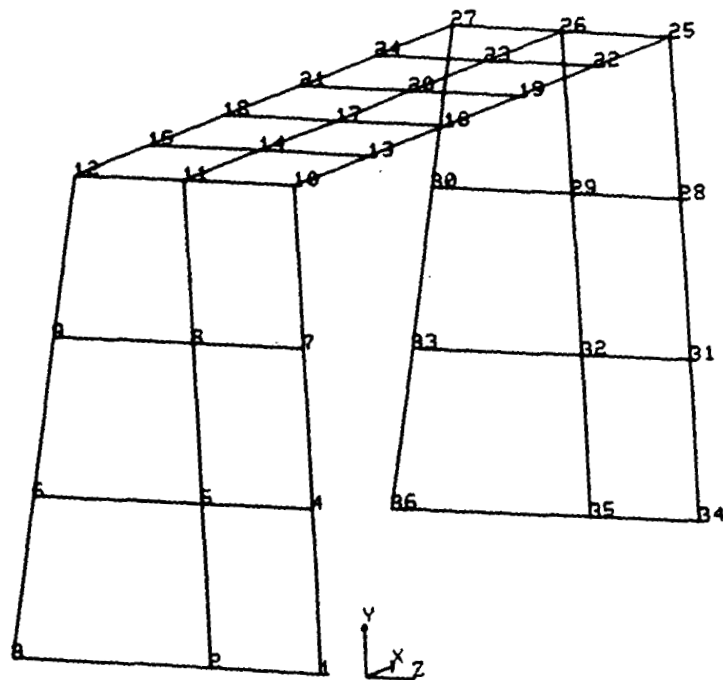


Figure 19.- Structure with optimized node numbering.

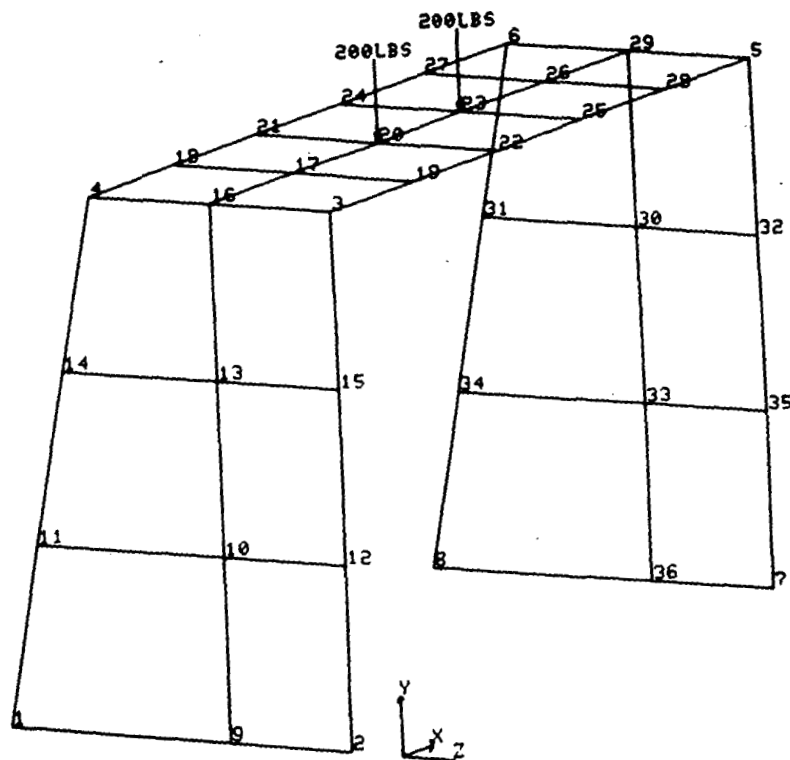


Figure 20.- Load specification with node number display option  
(200 lbs = 90.7 kg).

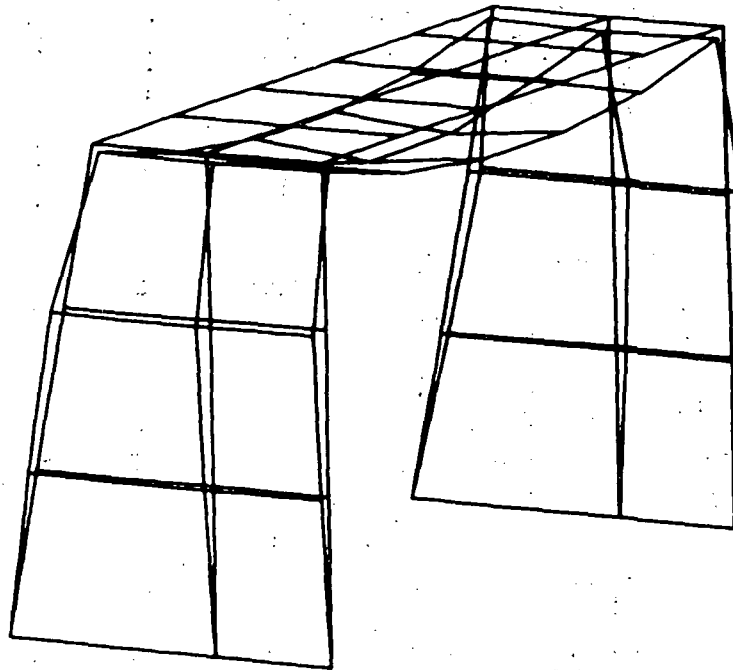


Figure 21.- Structure's deformation. Rotated view.  
Scale factor = 3.0.

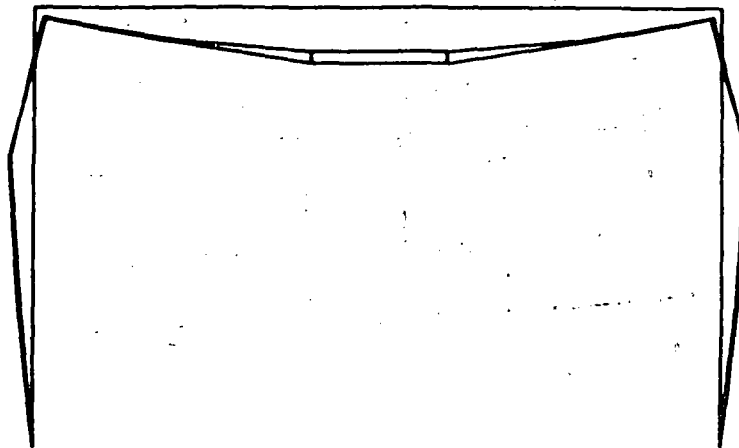


Figure 22.- Structure's deformation. Front view.  
Scale factor = 3.0.

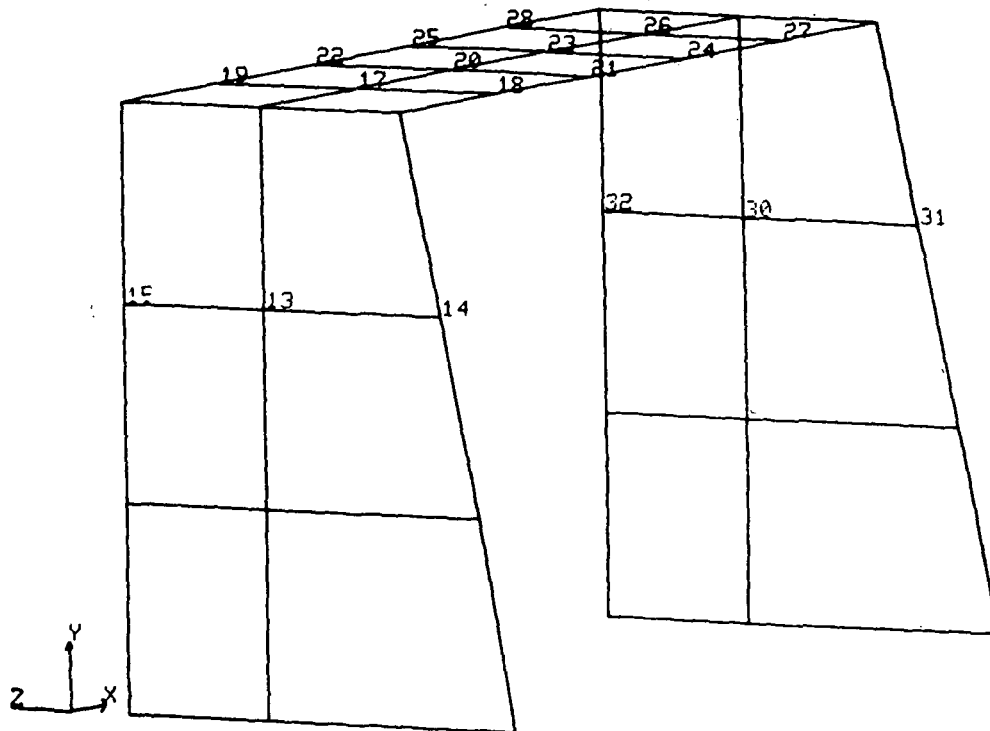


Figure 23.- Node point display for limit displacement (displacement above 0.025).

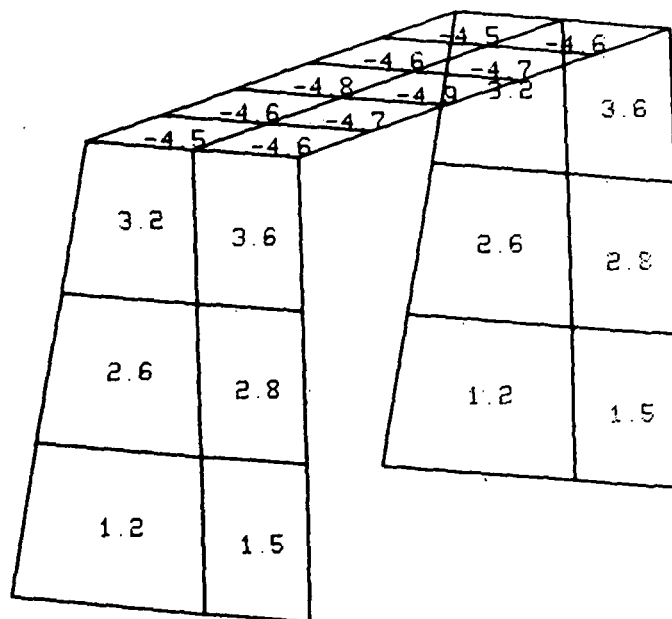


Figure 24.- Major principal stress (upper fiber, 6.85 MPa (1000 psi)) values on elements. (All may be displayed or only those above a specified value.)

-4.5	-4.6	-4.8	-4.6	-4.5
-4.6	-4.7	-4.9	-4.7	-4.6

Figure 25.- Rotated view of element stress plot, major principal stress (upper fiber, 6.85 MPa (1000 psi)).

**Page Intentionally Left Blank**

# THREE-DIMENSIONAL STRUCTURAL ANALYSIS

## USING INTERACTIVE GRAPHICS<sup>\*</sup>

Johnny H. Biffle and Hugh A. Sumlin

Sandia Laboratories

### INTRODUCTION

The design process for a structure consists of three phases: (1) the conception, from which a design drawing is produced; (2) the analysis, from which the integrity of the structure is determined; and (3) the redesign, if it is deemed necessary. When a three-dimensional analysis is needed, the time required to complete the analysis phase can be greatly reduced by the use of interactive graphics. The first area which requires a large amount of time is describing the structure geometrically to the analysis computer program when given a design drawing. When redesign is necessary, it becomes extremely important to be able to effectively redescribe the geometry of the structure for the analysis program. The second time-consuming area is the interpretation of the output from the analysis program.

When using the finite element analysis technique, the structure is subdivided into a finite number of distorted brick type elements or blocks; this results in the generation of a large amount of data which must be checked. The use of interactive graphics is an effective means by which to generate and check input data and to correctly and completely examine the large volume of output data. For a static analysis, the output data consist of the values of pertinent variables, such as stresses and displacements at each of the nodes (i.e., the corners of the elements) which are associated with the structure. For a dynamic analysis, velocities and accelerations are also output and a set of data exists for several times during the dynamic process.

If the structure is simple, without much curvature, batch processing is convenient and can be used effectively. However, for a complex structure, interactive graphics is a valuable tool. The analyst is provided the capability of not having to preselect the plots he wishes to see to examine the structure, as in a batch job. He can activate the job interactively and obtain the desired plotted data before leaving the interactive console. With the use of the Sandia Interactive Graphics System (SIGS), which consists of a Vector General cathode

---

<sup>\*</sup>This work was supported by the U. S. Energy Research and Development Administration.



ray tube, a DEC PDP-9 computer, a CDC 6600 computer, and an interactive graphics computer software system, both input and output can be effectively and efficiently processed. The hardware and software systems are described in more detail in the appendix.

### THREE-DIMENSIONAL INPUT PROCESSING

To describe the geometry for the analysis computer program, a mesh generation program is executed. Better analysis can be performed by using mesh generators because the structure can be divided into more elements than is practical by hand. MESH3D is a three-dimensional code (Ref. 1) which generates a mesh for which the input and logic is simple and straightforward, so it was selected as a basis for the interactive program.

In both the batch code and the interactive code, the process of generating a mesh consists first of selecting a rectangular set of elements which is to become a section of the three-dimensional mesh. The block of elements has I-elements in the  $I_x$  direction, J-elements in the  $I_y$  direction, and K-elements in the  $I_z$  direction (Figure 1). Twenty key points on the block are given  $x, y, z$  coordinate values; then the block is isoparametrically mapped into  $x, y, z$  space with the key point  $x, y, z$  coordinates controlling the mapping process (Figure 1). The structure is composed of a number of these building blocks. To examine the resulting generated structure, the three-dimensional representation must be plotted from several viewing directions. By examining the mesh, the analyst must decide if the geometry is defined correctly and if the size and distribution of the elements is what is needed for an accurate analysis.

With interactive graphics, the analyst can interact with the mesh generation program to edit data, plot when it is desirable to examine the mesh during execution, and re-execute after editing. Because the generator is modular in design, if he desires the analyst can plot after each operation or series of operations which generate a portion of the mesh. If there are errors in the mesh, the generating data can be displayed, corrected, and re-executed by interacting with the computer through the use of a light pen and a keyboard. Since all input errors in the initial data set can be corrected during the interactive session, the time consumed by resubmitting batch jobs to correct data is eliminated.

The ease with which the mesh generation code lends itself for use with interactive graphics is inherent in the concept of constructing the structure by assembling a number of building blocks. The operation for each portion of the structure and the block of elements which make up the portion can be completed before other parts of the mesh need to be generated for the rest of the structure.

The key words, DEFINE, MAP, MERGE, JOIN, COPY, ERASE, and UPDATE, are used to describe each operation involved in building the structure. For example, the divisions along the sides of a block are defined and then the block is mapped into x, y, z space. Another block can be defined and mapped; then the two blocks can be merged to form a new block. A list, such as DEFINE, MAP, MERGE, COPY, MERGE, etc., in the order the analyst wants the execution to follow is displayed on the screen. The analyst can then execute down the data, examining the results by plotting, change the data of a key word or add and/or delete key words and their associated data as is necessary. When the interactive session is over, the output consists of a set of data which can be used to execute the batch version of the mesh generator. If changes are necessary at a later time, the data can be used as input to restart the interactive program.

The surface of the structure is plotted on the cathode ray tube for examination. Rotation of the structure about its geometric center is accomplished by turning knobs on the control dials at the console, and the results are viewed instantaneously. Each knob provides the capability of rotating the body about an axis which is embedded in the body. After a desired view is obtained, as in Figure 2, a request can be made for a picture without hidden lines; the result is as shown in Figure 3. Either node or element numbers can be added to a hidden line drawing (Figure 4).

An example of building a structure is depicted in Figures 5 through 8. Figure 5 is a hidden line drawing of block 1, and Figure 3 is a view of block 2. A new block 1 is created by merging blocks 1 and 2 to obtain the block shown in Figure 6. A new block 2 is then created and merged with block 1 to obtain the structure shown in Figure 7. A rotated view of the block of element in Figure 7 is shown in Figure 8.

As has been mentioned, the design of the interactive graphics code is built around the batch version of the code. The program is overlaid and dynamically dimensioned, which means that only variables which need to be used are in core at the time of use. The data are kept in extended core storage (ECS) in the batch code but are put on a disk file with a data manager for the interactive code. However, all the interfaces with the data manager are in the same location as they were in the batch code, so conversion to ECS can be done efficiently when it becomes available to the graphics system.

A schematic of the organization of the PDP-9 and CDC 6600 function is shown in Figure 9. The commands EXECUTE, EDIT, PLOT, RESTART, ADD, DELETE, or TERMINATE are initiated at the PDP-9 machine by the use of a light pen. Control is then transferred to the CDC 6600 where the overlay CONTROL calls the appropriate overlays, such as DEFINE or BLDAR, to accomplish the requested task. When the task is completed or there is an

error, control is passed to the PDP-9 for further action, with the exception being TERMINATE for the end of the interactive session. Errors, when recognized, are returned through the overlay BURP to the PDP-9 and displayed on the CRT.

When the interactive job is initiated, the CDC 6600 program starts with overlay PROSTRT to initialize the location and size of arrays which are needed in blank common. Then control is passed to the overlay CONTROL where overlay BLD reads an initial set of data from a disk and stores the data on a random disk file through the use of a data manager. All data throughout execution are carried on a disk file, and only data which are needed for an operation are made available in core through the use of the data manager. Next, CONTROL calls overlay BILDISP which builds the main menu display with the key words of the initial data set and the PDP-9 commands. The main menu is sent to the PDP-9, and the interaction begins.

The main menu, with which the analyst works to activate the PDP-9 commands, consists of the commands and, as mentioned in the discussion of the key words, a key word list which represents the data necessary to build the mesh. If EXECUTE is hit with the light pen, the key word down to and through which the analysis is to be executed is hit. Control returns to the CDC 6600, and when the execution is completed or an error is detected, the main menu is displayed on the CRT and control is passed back to the PDP-9. Error messages are written on the display using the overlay BURP in the CDC 6600 program.

If EDIT is hit, the key word of the data the analyst wishes to edit is hit. Control is passed to the CDC 6600, and it creates an interactive display containing the data of the key word. Control then passes to the PDP-9, and the data are then displayed and corrected by the use of the light pen and keyboard before control is passed back to the 6600 to obtain the main menu.

If PLOT is hit, the analyst is asked what block he wishes plotted and then the CDC 6600 creates a display file for the plot. When control is returned to the PDP-9, the plot is displayed and can then be rotated. When the analyst is through examining the plot, the main menu is again displayed.

The command RESTART is used when the analyst wishes to reallocate blank common. The structure of blank common is such that the analyst must define in advance the number of nodes and elements in each block 1 through M where M is the total number of active blocks of geometry at any time. The overlay PROSTRT is used to accomplish the task, as in the beginning of the execution of the program.

If for some reason the analyst wishes to add a complete set of data including its key word, he hits the PDP-9 command ADD. A menu of key words is displayed from which to choose. With the light pen he hits a key word in the data set on the display just above where he wishes the data to be placed. The CDC 6600 then returns a display for the key word with a set of blank data which the analyst fills in before the data are inserted in the data base.

To delete a complete set of data and its key word, the command DELETE and the key word which represents the data are hit, causing the data to be deleted from the data manager file at the CDC 6600.

TERMINATE is used to end the interactive session. The corrected set of data is taken off the data manager file by overlay WRTOUT and written in a coded format on a disk. The set of data can then be used either to restart another interactive session or as input to the batch version of the mesh generator. A list of coordinates and node numbers which defines the structure located in block 1 is written to the OUTPUT file by the overlay OUTER, and the CDC 6600 execution is terminated.

If for any reason there is an abnormal termination of the job through an arithmetic error at any time during the interactive session, a recovery is made and overlay WRTOUT is called to create a set of input as it stood at the time of the error. The recovery was added to prevent loss of corrections the analyst had made to his mesh up to the time of the error. A new CDC 6600 job can immediately be activated and the analyst can continue to construct the mesh.

### THREE-DIMENSIONAL OUTPUT PROCESSING

As with the three-dimensional mesh generation there exists a batch code (PLTZ, Ref. 2) which is capable of performing all the needed tasks for output processing of three-dimensional data from a finite element code.

To display output data on a plane which cuts the structure, the plane is defined and then the following options are available.

Option 1 -- A plot of the intersection of the mesh with the cutting plane, showing both interior and exterior mesh lines.

Option 2 -- A plot of the intersection of the exterior boundary of the mesh with the cutting plane.

Option 3 -- A contour plot of the values of a variable (stress, strain, temperature, etc.) within the body on the cutting plane. The program accepts values of a variable

at each node to use as data for contour plotting. The exterior boundary as in option 2 is also plotted.

Option 5 -- A plot containing option 2 with the magnified displacements superimposed.

For example, the three-dimensional object shown in Figure 10 can be cut with a plane to produce a deformed shape (Figure 11); or option 4 could be used to produce a contour plot of a variable (Figure 12). It is also possible to define x and y limits so that blowups of a plot can be obtained (Figure 13). In the interactive mode, all this is accomplished by merely editing a small amount of card data for the batch code or the data which appear on the screen with the light pen and keyboard for the interactive program. After editing the data in the interactive mode, the plot is drawn in approximately 5 to 10 seconds. While sitting at the console, the analyst can view as many plots as he deems necessary to understand the output.

The CDC 6600 computer code is organized in the same manner as the batch version is organized. There is a main program which instead of reading card data retrieves a set of data from the PDP-9 to execute to produce a display file. The file is then displayed on the CRT along with the control data which was used to generate the plot. If the analyst wishes another plot, he changes only the data necessary to obtain the next plot and then signals the PDP-9 with the light pen to activate the CDC 6600 program. The cycle is repeated until all the plots that are needed have been generated.

## APPLICATIONS

To date, several problems have been analyzed which have utilized the capabilities of the graphics computer programs. In the area of electronic component design, the stresses and displacements associated with the epoxy in which components are encapsulated are important under various environments. The isometric view of the mesh for a particular analysis and output processing plots are shown in Figures 10 through 13. The encapsulant experienced cracks during a gun firing environment, which spun the assembly about the Z-axis. The analyst determined the stresses and displacements with and without a supportive ring on the circumference of the encapsulant.

Figure 14 shows the three-dimensional mesh which was used in a static analysis of a component design. The structure is a container for an electronic component, and the load consisted of a 10,000 g inertial loading and a  $7.58 \times 10^6$  Pa pressure loading.

## CLOSURE

With the interactive computer programs, the analyst can effectively perform three-dimensional input and output processing. Future work will be done in the area of output processing to remove the restriction of specifying a cutting plane by using visualization of a plane cutting and structure. With a color CRT, output processing will be performed to view deformed shape isometric plots and to view the contours of variables on the surface by using intensities of different colors.

## APPENDIX

### INTERACTIVE GRAPHICS SYSTEM HARDWARE AND SOFTWARE

The SIGS hardware, which is shown schematically in Figure 15 and pictorially in Figure 16, consists of a CDC 6600 computer, a DEC PDP-9 computer, and a Vector General 3D2 display. The CDC 6600 is used to generate all the Vector General display commands using Sandia-coded FORTRAN callable display generation routines. The 24K memory, DEC PDP-9 computer services the display interrupts and provides the necessary "bookkeeping" for the 6600-generated display file. Each of the Vector General displays is equipped with an alpha-numeric keyboard, a light pen, a character generator, display hardware subroutining, a 21-inch high-speed cathode ray tube, picture label scaling, intensity modulation, control dials, data tablet, and an interface to the PDP-9 through a direct memory access multiplexer.

CDC's Interactive Graphics System (IGS) provides the software interface to CDC's SCOPE operating system. At the remote terminals, the PDP-9 computer interfaces with the IGS through a Sandia-coded executive device handler (DPB). DPB handles all code translation, format conversion, communication synchronization, input-output file transmission, data transmission, and display file transmission.

A Sandia-code device handler (VGI) services all Vector General display interrupts, starts and stops the display, and allows the display file to be modified and manipulated. The Sandia-coded display manipulative routines are PDP-9 FORTRAN callable routines which permit the user to interact with or change all or part of the display using the light pen and/or keyboard.

## REFERENCES

1. Hutula, D. N., and Zeiler, S. M., "MESH3D: A three-Dimensional Finite Element Mesh Generator Program for Eight-Node Isoparametric Elements," Bettis Atomic Power Laboratory, Pittsburgh, Pennsylvania, AEC Research and Development Report WAPD-TM-1079, March 1973.
2. Hutula, D. N., and Wiancko, B. E., "MATUS: A Three-Dimensional Finite Element Program for Small-Strain Elastic Analysis," Bettis Atomic Power Laboratory, Pittsburgh, Pennsylvania, AEC Research and Development Report WAPD-TM-1081, March 1973.



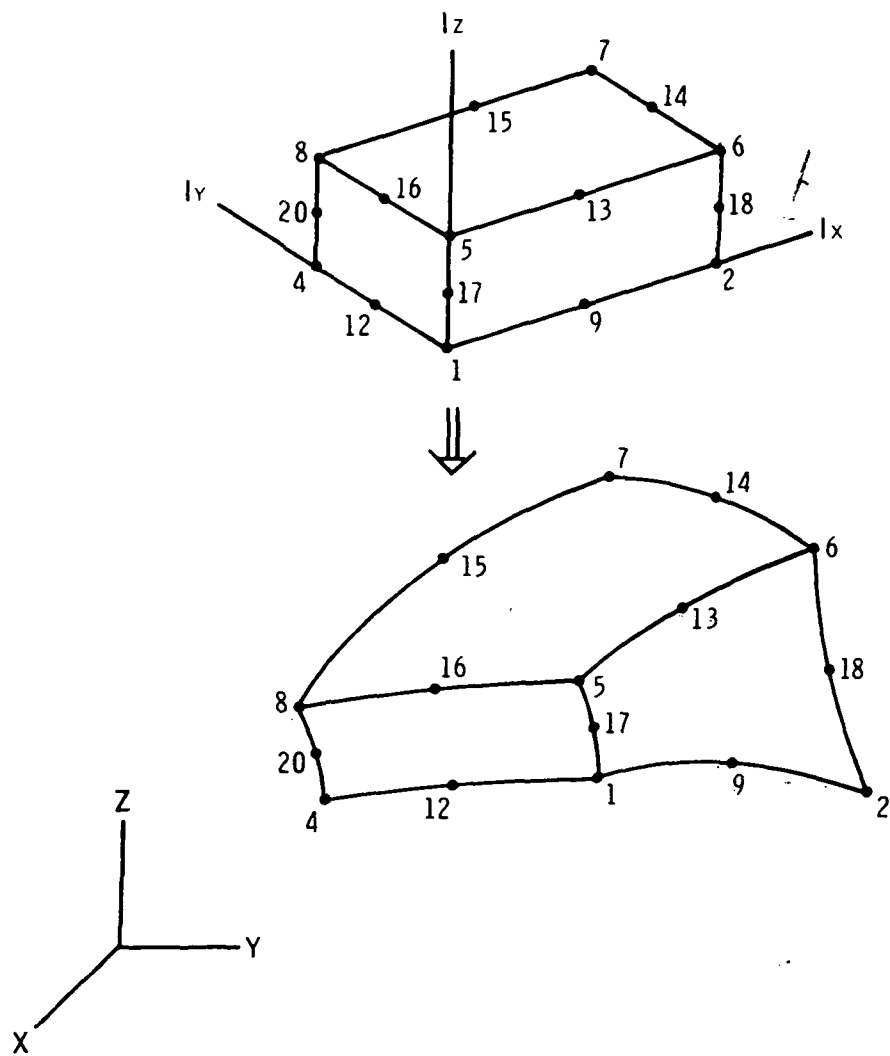


Figure 1. Mapping an integer block of elements into a curvilinear block of elements in  $x$ ,  $y$ ,  $z$  space.

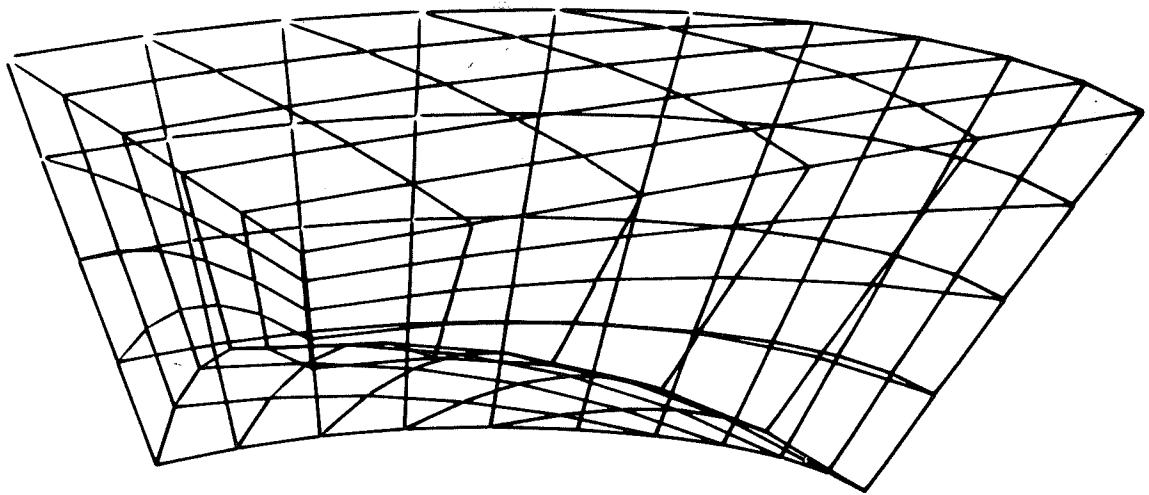


Figure 2. Rotated view of three-dimensional mesh.

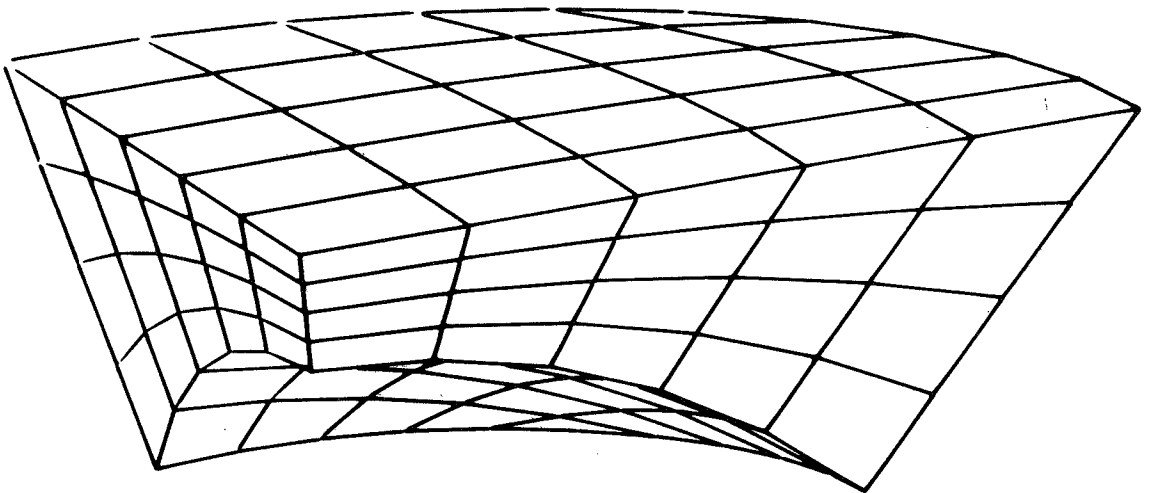


Figure 3. Hidden line view of three-dimensional mesh.

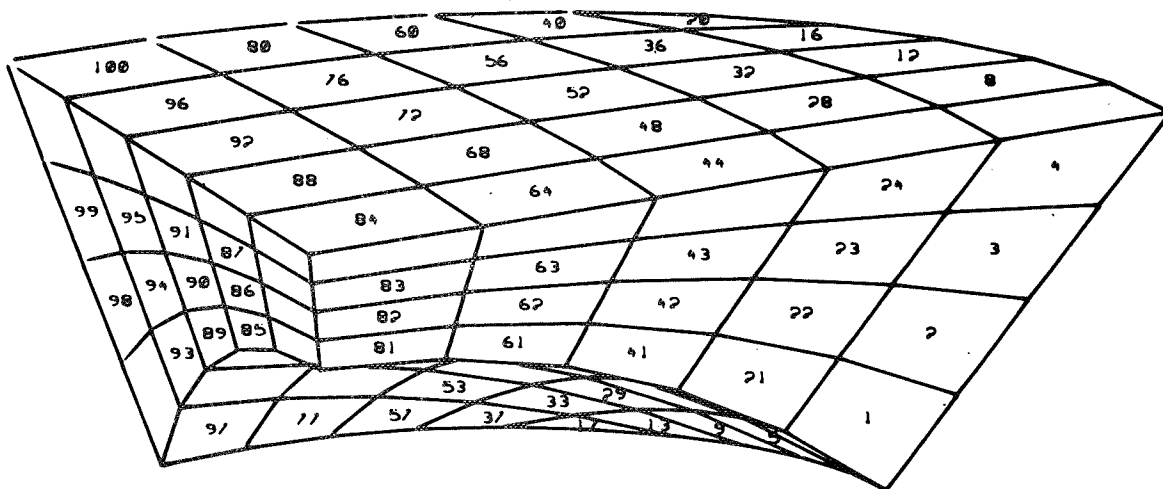


Figure 4. Hidden line view of mesh with element numbers.

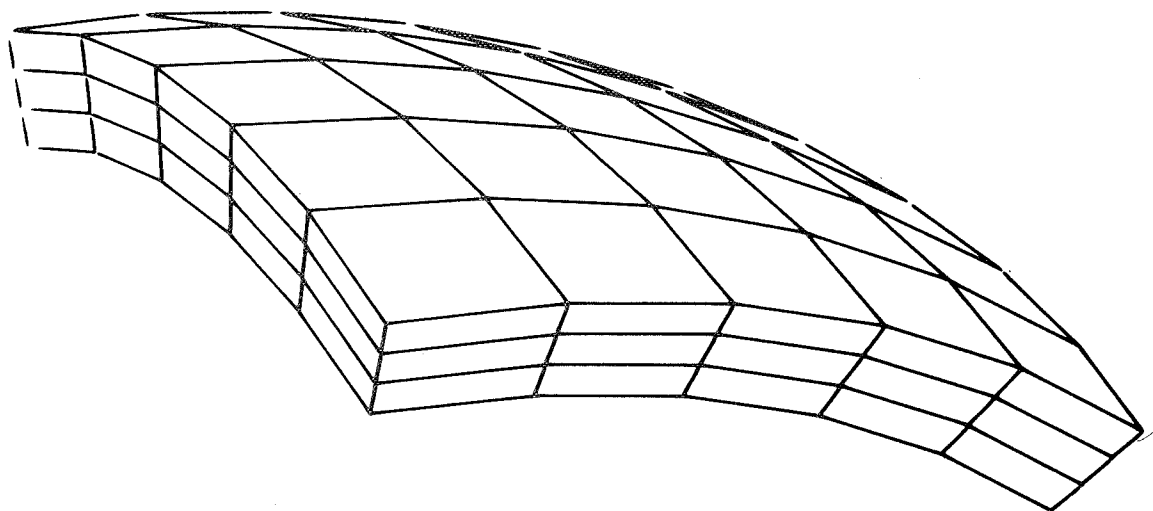


Figure 5. View of block 1 mesh.

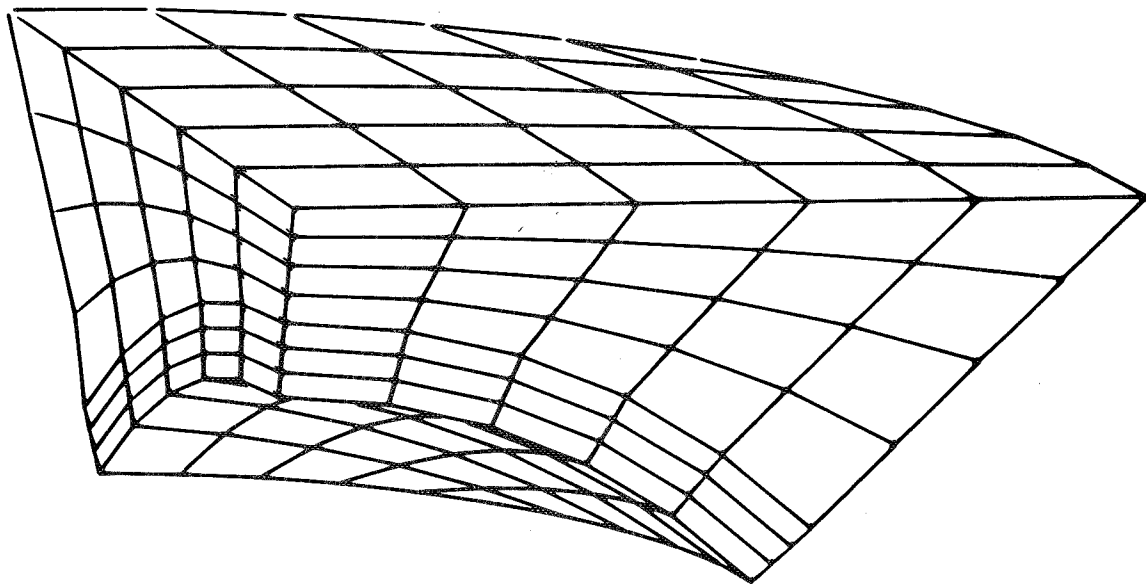


Figure 6. View of block 1 and block 2 merged.

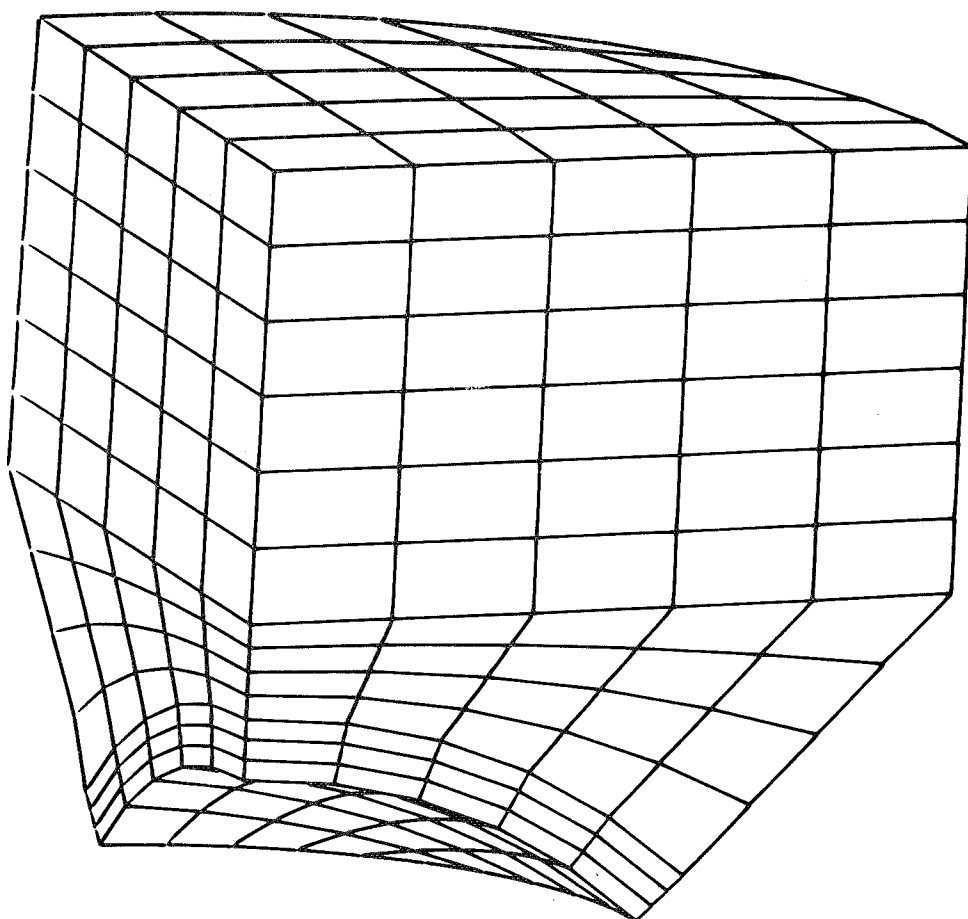


Figure 7. View of 3 blocks merged.

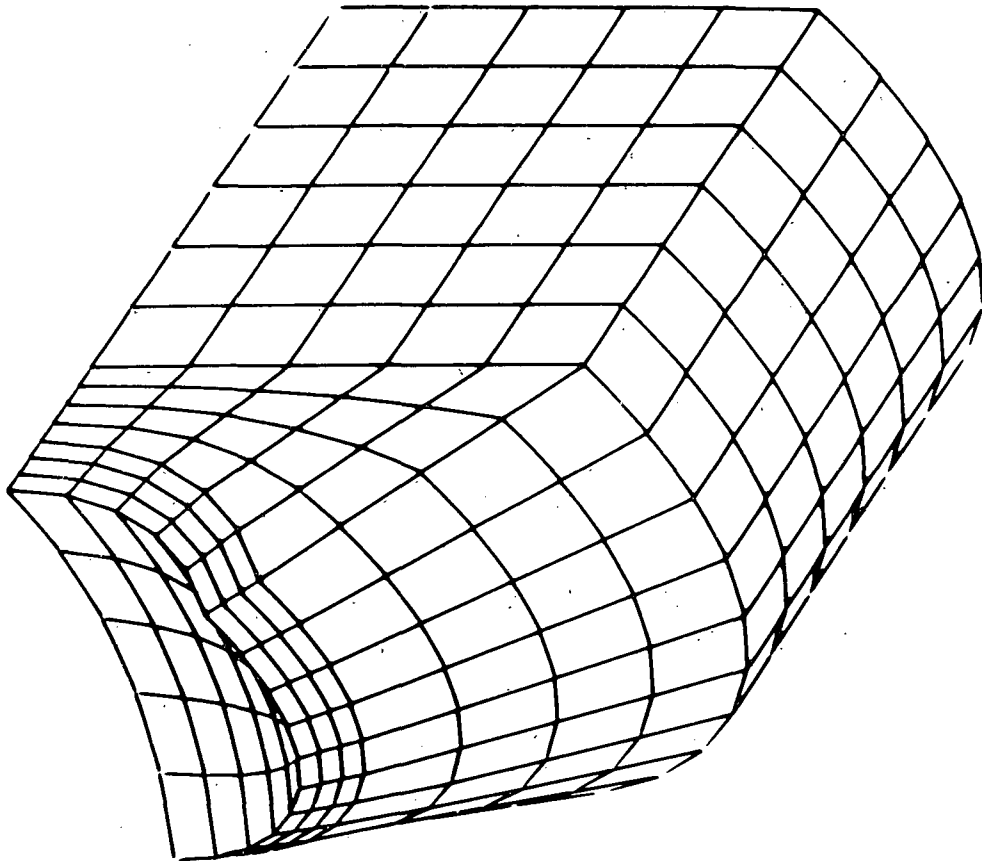


Figure 8. Rotated view of mesh.

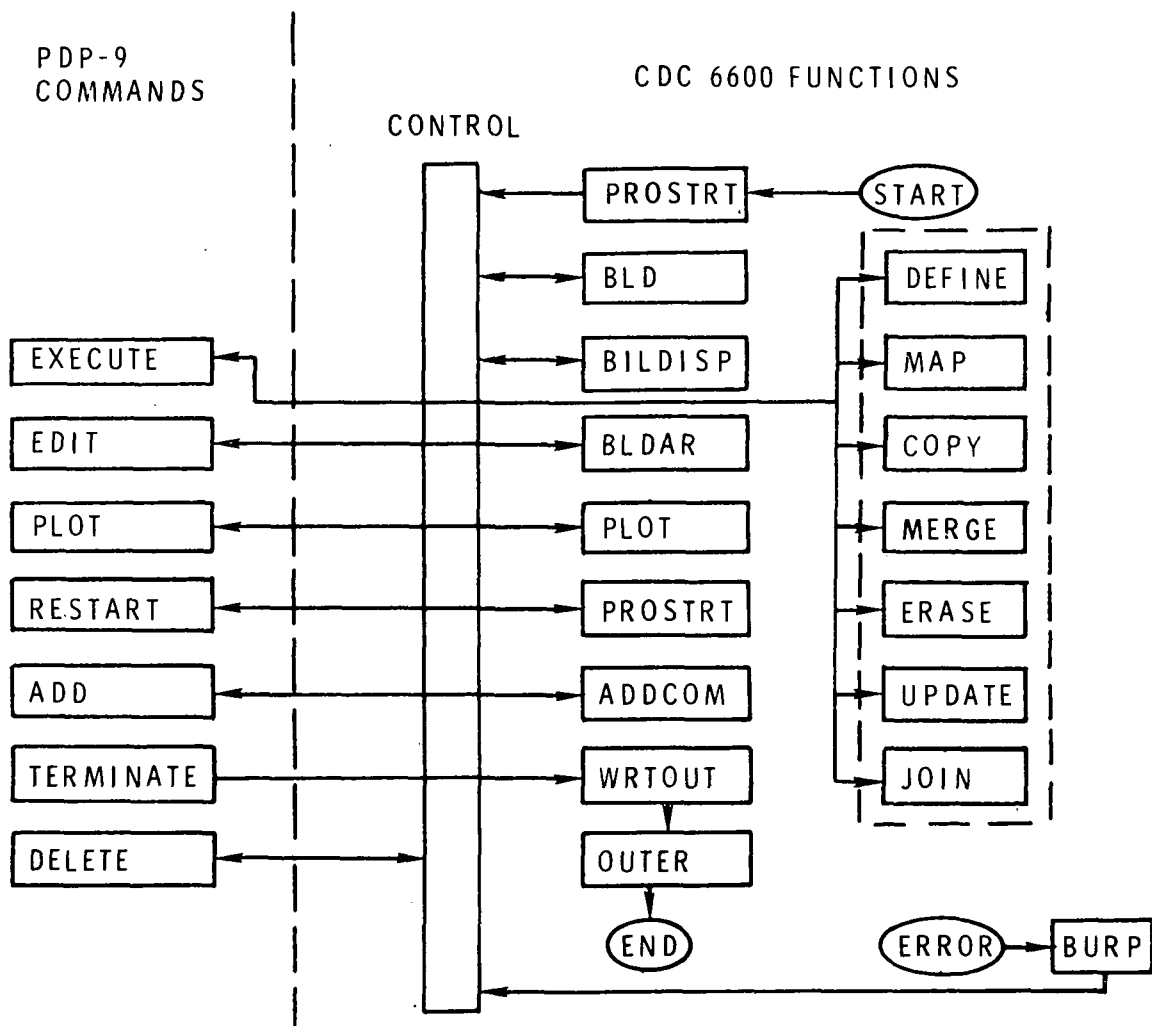


Figure 9. Interactive mesh 3-D computer code organization schematic.

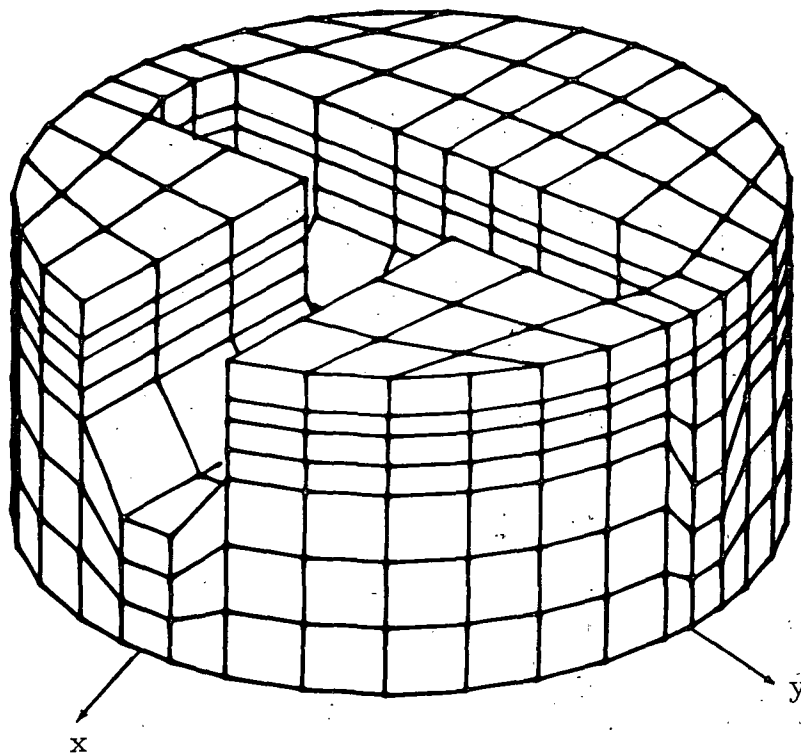


Figure 10. Three-dimensional view of encapsulant surrounding an electronic component.

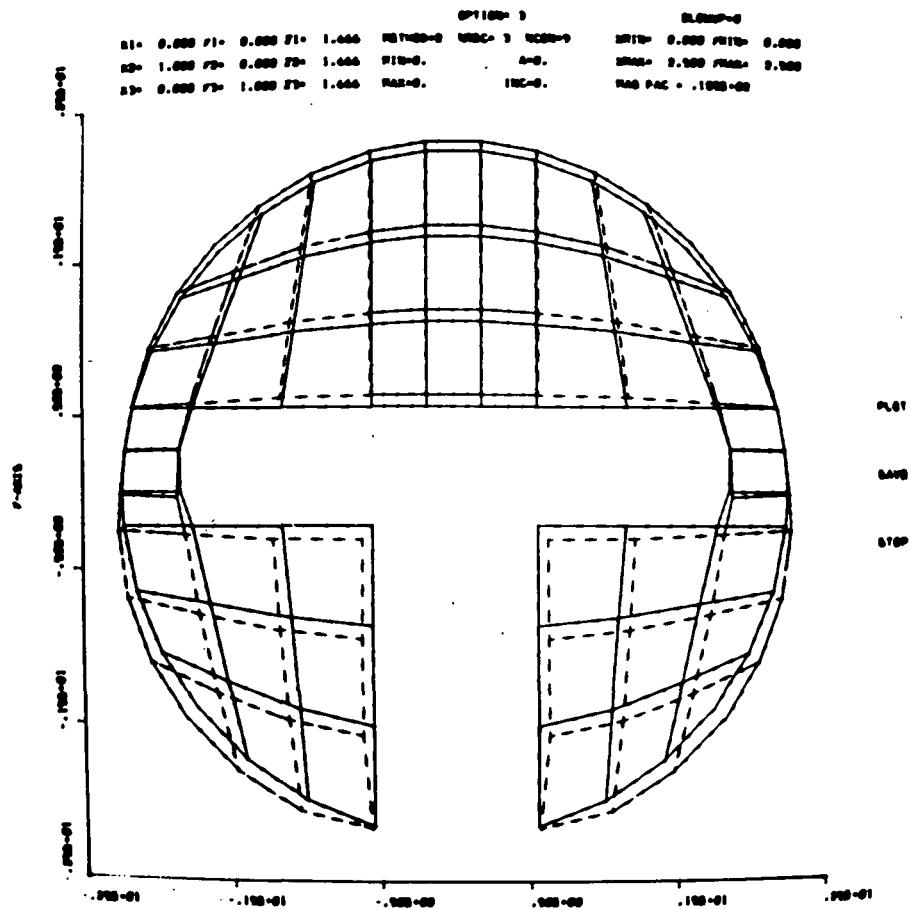


Figure 11. Deformed shape of component encapsulant in a spin environment.



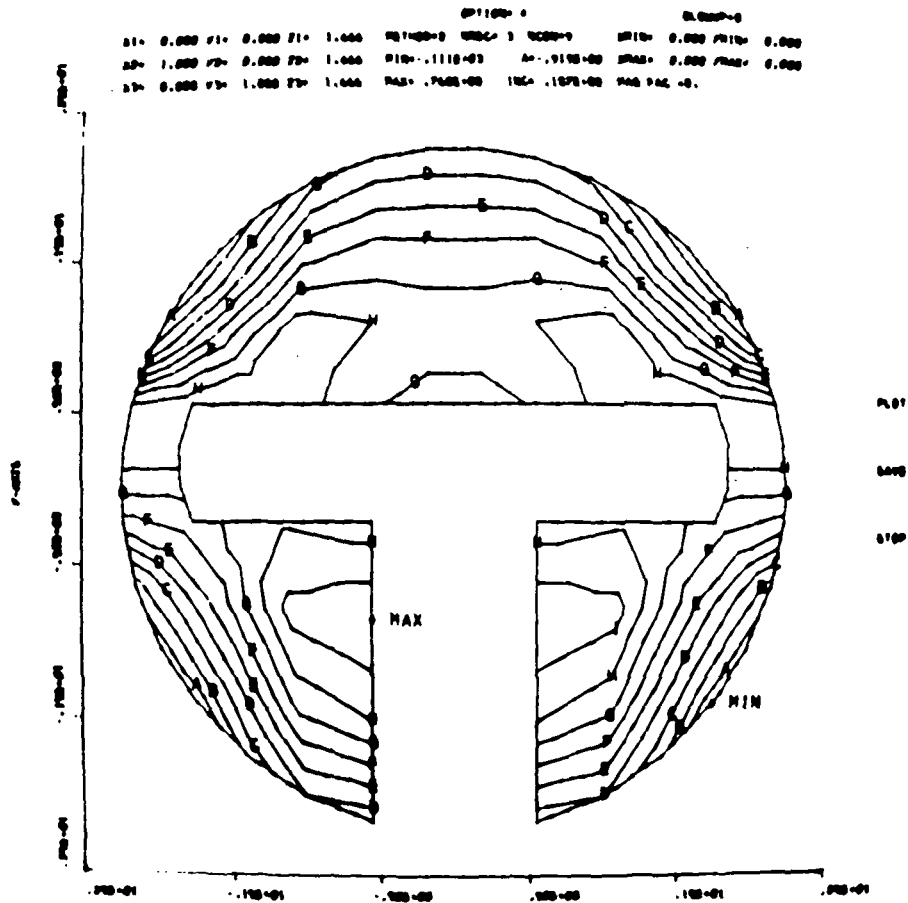


Figure 12. Contours of a stress component in a component encapsulation.



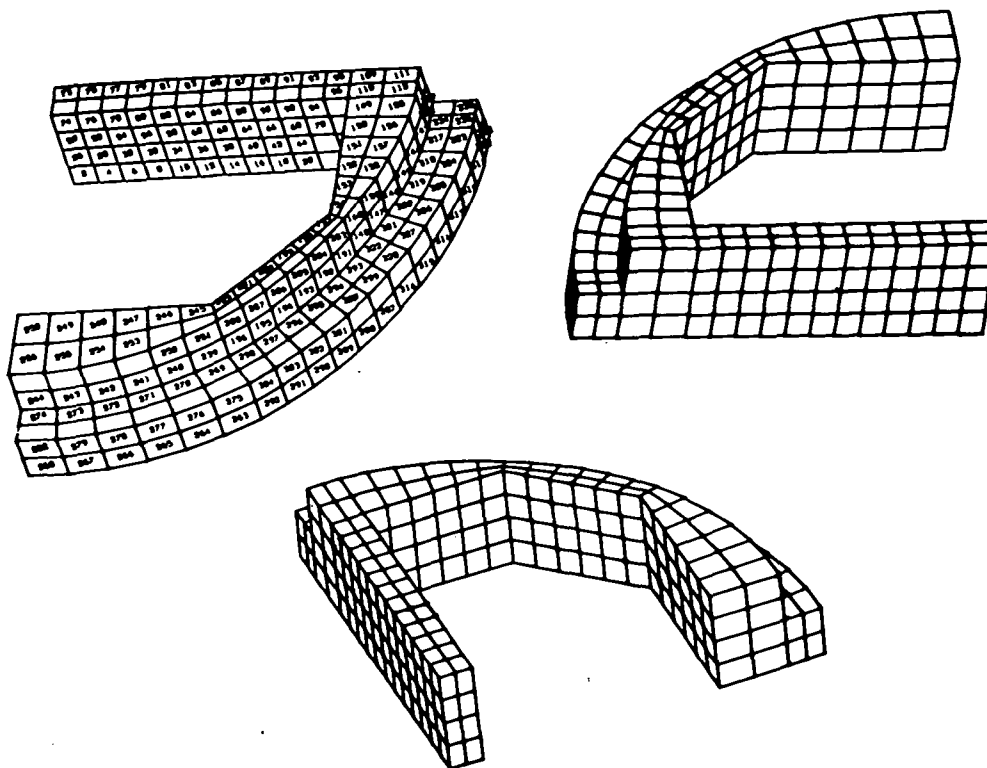


Figure 14. Three views of a mesh.

#### TYPICAL REMOTE CONFIGURATION

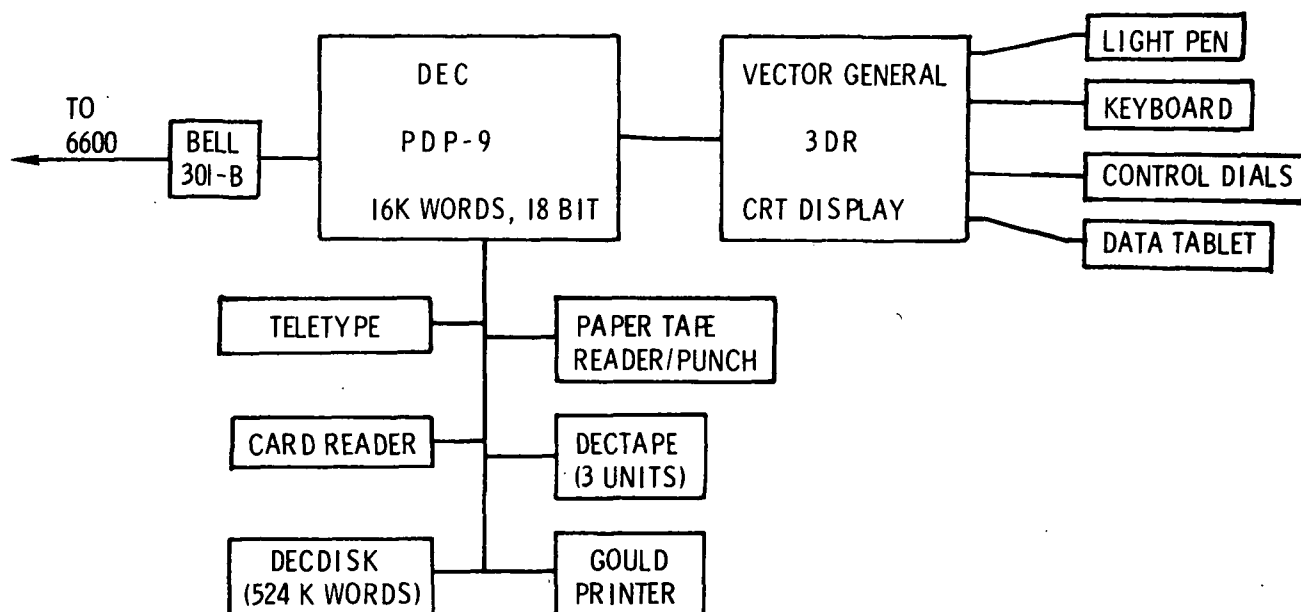


Figure 15. Schematic of interactive graphics hardware.



Figure 16. Interactive graphics hardware.

**Page Intentionally Left Blank**

## GRAPHICS FLUTTER ANALYSIS METHODS

An Interactive Computing System at  
Lockheed-California Company

Nick A. Radovcich

Lockheed-California Company

### SUMMARY

An interactive computer graphics system, Graphics Flutter Analysis Methods (GFAM) was developed by Lockheed-California Company to complement FAMAS, Lockheed's matrix-oriented batch computing system, and other computer programs in performing complex numerical calculations using a fully integrated data management system. GFAM has many of the matrix operation capabilities found in FAMAS, but on a smaller scale, and is utilized when the analysis requires a high degree of interaction between the engineer and computer, and schedule constraints exclude the use of batch entry programs. Applications of GFAM to a variety of preliminary design, development design, and project modification programs suggest that interactive flutter analysis using matrix representations is a feasible and cost effective computing tool.

### INTRODUCTION

The calendar time span required to transform aerodynamic data, airframe inertia, and structural design criteria into the form necessary for the structural design and sizing of the airframe has always occupied a large share of the total available time between concept and hardware. Releasing drawings to manufacturing before final definition of design loads and structural sizing are complete can lead to costly design and manufacturing changes.

The structural analysis process requires the interaction of seven major engineering functions: Loft, Design, Aerodynamics, Weights, Structural Analysis, Loads and Criteria, and Flutter. A sample of the organizational interfaces is shown in Figure 1.

Figure 2 illustrates Lockheed's present status and approach to computer-aided design for Structures and Aeromechanics. The Common Matrix Data is a data storage and access system common to FAMAS, NASTRAN, GFAM and auxiliary programs, such as aerodynamics transformation and safety margin. The data format is in the form of 2-dimensional arrays (matrices) which are identified by two 4-digit numbers, and, if necessary, the date the matrices were created. The computer automatically catalogs the identification when a new matrix is

entered into Common Matrix Data System; thus the computer is able to locate the matrix when requested by any one of the programs having access to the matrix data storage.

FAMAS (Flutter And Matrix Algebra System), a Lockheed-developed computing system, performs static and dynamic loads analysis, flutter analysis, plotting, response analysis, and a complete set of linear algebraic operations, such as matrix inverse, matrix eigenvalue problems, matrix multiplication, etc., in addition to standard non-matrix-algebraic operations such as element by element multiplication, store, extract, etc.

NASTRAN is a structural modeling and analysis program which performs vibration analysis and certain matrix operations found also in FAMAS.

Computer graphics application at Lockheed-California Company is divided into two functions: drafting package and analytics. The primary drafting package program is CADAM (Computer Augmented Design and Manufacturing), a powerful design tool used in lofting, drafting, and numerical control tasks. CADAM has its own data base and data management system and drives 7 or more scopes per 130K core partition.

Under analytics, a variety of user application programs are executed, each, when running, requiring a separate 126K core allocation. The ICSMP (Interactive Continuous System Model Program) package is used extensively in the simulations of the airplane for maneuver, landing, and impact analysis, by integrating a set of ordinary differential equations. A number of graphics programs assist aerodynamics and other disciplines. Interactive flutter analysis, formerly done using the Network Analyzer, a direct analog computer, is now available as graphics analytic program, GFAM.

This paper reports on the development and use of GFAM to accelerate the design process in satisfying the flutter requirements, especially when the design exhibits flutter deficiencies.

## DESCRIPTION

### General

GFAM, using a matrix data base generated for batch flutter analysis in the FAMAS system, performs interactive flutter analysis, structural optimization to satisfy flutter requirements, control synthesis for CCV (Control Configured Vehicle) applications, general matrix algebra operations, and the matching of structural dynamics analysis to ground vibration test data. GFAM supports test data correlation, flutter methods development, and quick analysis of a design for flutter and structural dynamic characteristics during preliminary and point design phases. GFAM applications will expand into a more extensive graphics generation capability of the stiffness and mass characteristics of the design from coordinate and physical property descriptions. Future discipline additions may include limited loads and

stress analyses. Terminal requirements and other pertinent data are tabulated in Table 1; GFAM's general computing capabilities are summarized in Figure 3, and the computing system location in the IBM 360 computer tree is shown in Figure 4.

#### GFAM Control Display

The engineer controls the operation of GFAM through a graphics display shown in Figure 5. Column one lists members containing 80-column data cards (image format) e.g., program code as listed in Appendix A, which are moved to the card input file, FT05, when the member name is detected by the light pen. The data in the input file can be modified, copied, merged, deleted, or saved as another member. The card image input file is accessible by programs listed in column three. Each of these programs which operates in the interactive mode is identified by an 8-character name and can be individually loaded and executed using the SUBMIT option.

Column two in the GFAM control display lists users' disc files which store matrix data accessible by programs submitted for execution. These data storage files are available while the problem solution is actively pursued. Matrix data may be moved by the user to tape or disc in the FAMAS system for long term storage or for access by the FAMAS system.

The following is a description of GFAM control display menu:

FETCH: Causes the cards in the detected member to be shown on the scope for viewing and editing.

SAVE: Saves a modified card data set as a new or old member in the FT05 column.

PRINT: Prints the active FT05 data set member.

PURGE: Purges a card data set from the FT05 column.

SUBMIT: Transfers control to one of the programs in the PROGRAMS column.

OUTPUT: Permits the review of print and punch output of the last SUBMIT. The user has the option to send the print output to the printer and to save the punch data set as another FT05 member.

LEAVE: Terminates a session at the scope.

#### Program MATRIX

MATRIX is a general matrix algebra computing program which is used to generate, condition, and manage matrix data requirements of the technology modules which will be discussed subsequently.



Each matrix algebra operation is performed interactively, and the average wait time at the scope for each operation is 10 seconds, with the maximum wait time designed to be not more than 2 minutes.

The following are examples of user-supplied coding that performs the more common matrix operations:

- (1) Create new stiffness matrix after changing the (3,3) element of the compliance matrix (see equation (8)).

C @ 3001	identifies matrix 3001 as C
CM @ 2001	identifies matrix 2001 as CM
C(3,3) = .006	modifies the (3,3) element of C
CI = INV(C)	generates the inverse of C
K = CM*CI*CM'	forms stiffness matrix
PRNT K,C	prints matrices K and C
RNAM K as 4000	reidentifies variable K as matrix 4000
KEEP 4000	moves matrix 4000 to user file

- (2) Perform static reduction to 100 degrees of freedom on a 200 x 200  $[K]$  matrix partitioned such that  $[K] = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}$ . The reduced stiffness matrix is defined by:

$$KR = K_{11} - K_{12} * K_{22}^{-1} * K_{21} \quad (1)$$

K22 = XTRC 100 BY 100 FROM K (101,101)

K12 = XTRC 100 BY 100 FROM K(1,101)

K11 = XTRC 100 BY 100 FROM K(1,1)

K22I = INV (K22)

KMD = K12 \* K22I \* K12'

KR = K11 - KMD

or by a macro instruction,

KR = GYAN (K) BY 100.

With the macro instruction,

KR = GYAN (K) BY V SAVE G

the program will shuffle  $[K]$  according to the vector V and output the Guyan transformation matrix  $[G]$  for use on a consistent mass matrix  $[M]$  where

$$[MG] = [G]^T [M] [G] . \quad (2)$$

(3) Perform a vibration analysis:

M @ 1002	identifies matrix 1002 as M
K @ 3003	identifies matrix 3003 as K
MI = INV(M)	forms inverse of M
A = MI*K	
ROOT = EIG(A)	extracts eigenvalues by QR
V = VECT(ROOT, A)	
FROM 1 to 30	extracts eigenvectors for the first 30 roots
RNAM V AS 5000	reidentifies variable V as matrix 5000
PRNT ROOT, V	Prints matrices ROOT and V
KEEP 5000	move matrix 5000 to user file

Appendix A includes another MATRIX program example with line by line explanation of the code.

The first MATRIX display with which the user interacts is a restart option from a previous checkpoint. The next display is the INPUT DISPLAY shown in Figure 6. From here the user may read matrix data from tape or disc/DATA READ/, write matrix data on tape or disc/DATA WRITE/, edit the user file/MATRIX TABLE DISPLAY/, return to the user program/PROGRAM/, and move the user program cards from the FT05 active file into MATRIX program file/CARDS/.

The user-supplied coding to be used by program MATRIX may be accessed with the/CARD/option, from the FT05 members saved previously, or the user may generate the coding directly in the PROGRAM DISPLAY shown in Figure 7. The user also controls the execution of the coding from this display and has the choice of executing the code singularly or in designated blocks. If the coding is incompatible with the indicated matrix operation requirements, an error code is given the user indicating the type of error and suggesting the required action for its correction.

Another function of MATRIX is data management of the matrices in the user files. Figures 8 to 11 illustrate displays used for reading data from tape (FAMAS) or disc into GFAM users' file. Figure 12 is a display that the user sees after a read attempt. Matrices may be easily moved from one user file to another.

Data write function is controlled by DATA WRITE DISPLAY shown in Figure 13. The user may write a FAMAS tape to save his matrix data indefinitely and make it available for batch programs using the FAMAS matrix format.

Matrix management and the associated identifiers of the user file is possible in the MATRIX TABLE DISPLAY shown in Figure 14, where matrices may be purged and the file rebuilt. The MATRIX TABLE DISPLAY lists the number of matrices, KOUNT, on the user file; the matrix number, e.g., 9090; the number of rows of the matrix; number of columns; the type (real or complex); the logical file on which the matrix is stored, FILE; the relative position on the file, POS; and three identifiers, FREQ, DAMP, and SET#. FREQ and DAMP are two numeric values associated with each matrix on the file. For example, all aerodynamic matrices have a value of reduced frequency associated with the matrix. By convention, the value of the reduced frequency is stored under the FREQ identifier, e.g., matrix 1863 has a value of .760 for reduced frequency in Figure 14. Values of FREQ and DAMP can represent almost anything the programmer/engineer decides is helpful when the matrix is being used in a technology module. The identifier, SET#, is used to group matrices together as one set. In Figure 14, matrices 1863 to 1873 all have the set number 71. In the FLUTTER program, the user needs only to identify the aerodynamics by the SET#, i.e., 71. The program scans the user file and pulls into the program all matrices with a SET# equal to 71. In addition, program FLUTTER scans the FREQ identifier of matrices with SET# equal to 71 and uses FREQ values as reduced frequency. Once the values of the identifiers are part of the MATRIX TABLE they are available to programs such as FLUTTER, OPTX50TH, F.VEL, etc. (Figure 3), without any additional action on the part of the user. Since these identifiers are an integral part of GFAM's matrix data set, program MATRIX provides three ways to generate identifier values for the MATRIX TABLE. First, the READ WITH LIST display, Figure 9, generates the identifiers by indexing DELTA FREQ and DELTA DAMP for the matrices in the read list. Second, if the list of identifiers is used many times, a matrix of these identifiers generated previously and stored under a separate matrix number can be referenced in the FREQ/DAMP GROUP ID. Third, in MATRIX TABLE display, Figure 14, the identifiers can be generated or changed directly.

Once the user has built the MATRIX TABLE, the information need never be entered into the computer again. When the user wishes to save his file on tape, he has the option to save the MATRIX TABLE by assigning a matrix number to #FOR MAT MATRIX in Figure 13. When the matrices are written out on tape, a separate matrix is generated that saves all the identifiers, as well as the matrix numbers, that the user identified to be saved. To load the file with the data saved on tape, the user enters the MATRIX TABLE#, light pen detects /READ/or/READ & KEEP/ in Figure 10, and all the matrices together with all identifiers will be loaded into the user file. This permits the use of one

file by many part-time GFAM users with this simple data storage and retrieval system.

### Flutter Programs

There are four basic flutter programs, FLUTTER, F.VEL, FLUTFEED, and OPTX50TH, to assist the engineer in performing interactive flutter analysis once the matrix data describing the mass, stiffness, aerodynamics, and CCV properties are available.

The flutter matrix equation,

$$[D(p, V, \bar{M})] \{\bar{z}\} = \{0\} \quad (3)$$

represents the linearized equations of motion for a flexible structure with unsteady aerodynamics. Program FLUTTER solves the characteristic equation

$$| [D(p, V, \bar{M})] | = 0 \quad (4)$$

for a non-dimensional root,  $p$ , when the airplane velocity,  $V$ , and the Mach number,  $\bar{M}$ , are specified. If the air density replaces the Mach number as an independent variable, solutions are made at different  $\bar{M}$ , but the aerodynamic matrices are not adjusted for Mach number effects. Equation (3) was formed by substituting  $\{z\} = \{\bar{z}\} e^{pt^*}$  into the equations of motion, where  $t^* = \frac{tV}{\bar{c}}$ ,  $\bar{c}$  is the characteristic length, usually the semi-chord of the airplane, and  $\{\bar{z}\}$ , the eigenvector, is proportional to the initial conditions required to excite this mode only. The eigenvalue,  $p$ , has the form,  $p = \gamma + ik$ , where  $i = \sqrt{-1}$ ,  $k$  is the reduced frequency, and  $\gamma$  is a non-dimensional real part indicating the mode stability - stable if  $\gamma$  is negative, unstable if  $\gamma$  is positive, and neutrally stable (flutter point) if  $\gamma$  is zero. The exponential form of stability or instability results from  $\{z\} = \{\bar{z}\} e^{\gamma t^*} e^{ikt^*}$ .

Another form of solution to the equations of motion is the standard k-method, where  $g$  is the structural damping required for neutral stability. The relationship  $g = 2\gamma$  provides the representation of  $p$ -type solutions of equation (3) in the  $g$  format of the standard k-method solution.

FLUTTER computes f-V-g (frequency, velocity, damping) as shown in Figure 15 for any in-flight mode. The user, however, controls the computations through the OUTPUT display so that only the modes and velocities of interest are computed. Damping versus velocity for each mode is plotted in the OUTPUT display as shown in Figure 16. The user can purge roots from the solution /POUT/, review purged roots /ROLL/, store solution roots of interest /STORE/, checkpoint the computations up to that time /CKPT/, view past solutions numerically /PASTSOL/, and select the best input trials for new root computations from the past solutions /TRIAL/.

Program F.VEL directly computes the flutter speed (damping = 0) and the associated frequency.

Program FLUTFEED computes flutter roots when CCV and autopilot equations are added to the basic flutter equation. The data for the transfer functions and matrix numbers of the associated sensor matrices and the force distribution matrix are stored under one matrix number. Figure 17 is the INPUT display, where the transfer function data may be entered together with matrix information for mass, stiffness, etc., if this is not already available through an FTO5 member data set. If the transfer functions need to be changed, the engineer enters the index of the matrix to be modified and light pen detects /MODIFY/. The display shown in Figure 18 gives the transfer function data in the form of polynomials,

$$\prod_{i=1}^4 \left( \frac{a_0 + a_1 s + a_2 s^2}{b_0 + b_1 s + b_2 s^2} \right)_i \quad (5)$$

where  $s$  is the Laplace operator. If elements of a force distribution matrix require new values, the user need not transfer out of FLUTFEED and enter MATRX, but simply light pen detect /MODIFY DELTA/, and, as shown in Figure 19, change the matrix elements.

FLUTFEED OUTPUT display is similar to FLUTTER, except for additional displays with which the engineer determines the phase and gain between a control surface and a sensor which satisfies a prescribed damping and frequency requirement.

Finally, program OPTX50TH in GFAM resizes the structure to satisfy the flutter constraints and to keep the weight increase at a minimum. The program assumes the mass and stiffness matrices to have the form

$$[M] = [M_o] + [\Delta M_i (\beta_i)] \quad (6)$$

$$[K] = [K_o] + [\Delta K_i (\beta_i)] \quad (7)$$

where  $[M_o]$  and  $[K_o]$  are the base mass and stiffness matrices, and  $[\Delta M_i]$  and  $[\Delta K_i]$  are delta mass and stiffness matrices associated with the  $i$ th design variable  $\beta_i$ . The delta stiffness matrix is a function of  $\beta_i$  up to the third power, while the delta mass matrix is only linear in  $\beta_i$ .

## VEHICLE DESIGN FOR FLUTTER

### General

When a design is deficient in satisfying the flutter requirements, the flutter analyst in conjunction with representatives from other disciplines defines candidate areas for design changes. The magnitude of design changes

required to satisfy flutter constraints is determined by analysis, and the design with flutter changes is recycled through stress and possibly loads, as shown in Figure 20. The loads-stress-flutter cycle is an integral part of preliminary design and project design. The simplified structural models for preliminary design permit much shorter analysis cycle time than is possible in project design.

### Structural Resizing For Flutter

The general analysis flow for structural resizing and/or ballasting of a flutter deficient design for minimum weight addition is shown in Figure 21. An overview of GFAM's role during this task follows:

Aerodynamic matrices  $[\hat{A}]$  are usually available from the baseline flutter analysis. The computation of  $[\hat{A}]$  is presently and, in the foreseeable future, will remain a batch function. However, the generation of the batch program input data may become in the near future a graphics function in GFAM.

Transformation matrices,  $[D_x]$  and  $[D_z]$ , used to transform structural deflections into local angles of attack associated with the aerodynamic loads points are presently generated in batch and are part of the aerodynamic matrices  $[A]$ . Because the baseline flutter analysis must precede any resizing task, the goal is to make generation of  $[D_x]$  and  $[D_z]$  a GFAM task to reduce the overall elapsed time.

Structural model data generation in GFAM at present is restricted to a simple structural model representation. It includes the formulation of the connecting matrix  $[CM]$  and the diagonal compliance matrix  $[C]$ , which are used to form the stiffness matrix,

$$[K] = [CM] * [C]^{-1} * [CM]^T \quad (8)$$

Equation (8) is computationally efficient when the sparseness of  $[CM]$  and the diagonality of  $[C]$  are taken into account in the triple matrix multiplication. Stiffness derivative matrices required by the structural optimization program are readily generated from equation (8). Program COMPLY in GFAM computes the diagonal compliance matrix from physical properties of the structure.

When the stiffness matrix cannot be represented by equation (8), the stiffness matrix computation must be performed in batch using NASTRAN. A NASTRAN pre-processor using the scope interface is currently under study.

There are presently no program modules to assist the engineer in forming the mass matrix from distributive data. However, the mass matrix is often formulated as

$$[M] = [DM]^T * [ME] * [DM] \quad (9)$$

where  $[ME]$  is a diagonal elemental mass matrix. This formulation offers the required flexibility to generate design variable mass derivative matrices.

Vibration analysis is usually preceded by a static reduction of the stiffness matrix (equation (1)) and a Guyan reduction of the mass matrix (equation (2)) to reduce the problem size down to 100 degrees of freedom for GFAM or down to 130 to 200 degrees of freedom for batch processing using FAMAS. If the matrices are a function of both static reduction (equation (1)) and vibration modes, then updating the matrices for these two effects is a necessary part of the structural resizing procedure as the structure mass and stiffness characteristics are being modified.

Baseline flutter f-V-g curves are usually computed in batch. When the design is altered and the engineer wants to determine the effect of the design change on some f-V-g curves, program FLUTTER in GFAM solves equation (4) for p and plots the roots as a function of velocity. The program tracks the solution of a single mode through the velocity range requested by the engineer.

Structural resizing occurs in two parts. First, is the initial structural resizing to satisfy all flutter requirements. Second, is the resizing for minimum weight while explicitly satisfying the flutter requirements and not violating strength requirements. The engineer performs both resizings using program OPTX50TH in GFAM. Details of the first resizing may be found in Chapter 8 of Reference 1, while details of the optimization are reported in References 2 and 3.

#### Active Control Technology

The current application of ACT methods is maneuver loads relief, gust loads alleviation, fatigue life increase, and ride quality control. The engineer synthesizes a feedback or feedforward control system by prescribing in-flight damping for certain modes as a function of velocity, as shown in Figure 21, and/or flight regime.

The prescribed damping curves result from studies in which the sensitivities of gust loads, etc. to changes of the in-flight damping of different modes are determined. Different combinations of control surfaces and sensor locations are evaluated with respect to the prescribed damping curves. The control synthesis process is optimized to derive the greatest benefits to the ACT goals in terms of matching the prescribed damping curves. The basic tool in this evaluation is program FLUTFEED in GFAM. Its main task is the computing of exact gain/phase relationships required between a control surface and a sensor for the prescribed damping and frequency of a given mode.

#### PRELIMINARY DESIGN EFFORT

A typical preliminary design effort involves the formulation of a structural model using a simple Russell Beam or 2-dimensional (2-D) representation with 300 or less structural elements. Typically, the preliminary design group initiates a new design from which the stress group sizes the proposed aircraft configuration using strength criteria. A structural model is



formulated, together with the aerodynamics and inertia matrices, and a standard vibration and flutter analysis is performed. A full flutter f-V-g plot is determined, and the design is subjected to Mach number and weight configuration variations. Typically, the design will be flutter deficient, because the sizing is primarily based on strength. The task is then to size the structure for flutter and determine the sizing changes that must be made to the design to satisfy the flutter constraints. The matrices associated with the mathematical model are loaded into the GFAM files by accessing the FAMAS tapes (disc). The engineer then defines the candidate sections of the structure which would be most effective in removing the flutter deficiencies. Because the structural model will typically have many more structural elements than needed for independent design variables, the practice has been to assemble a number of sizing elements which may be uniformly changed into specific groups. Since the best groupings of sizing elements are not known a priori, the engineer starts with design regions of arbitrary size and interrupts the solution process if the data indicates that groupings of structural elements are too coarse or too fine. The weight penalty due to stiffness requirements of the structure must be given to the group evaluating the proposed design.

An alternative to changing structural sizing to alter the design flutter characteristics is to add mass ballasting. In structural optimization, mass ballasting is simply another design variable with  $[\Delta K_i (\beta_i)]$  in equation (7) equal to zero.

During the resizing task, the engineer periodically checks dynamic characteristics of the design by taking the current mass and stiffness matrices and initiating a flutter analysis to verify that the resizing analysis is tracking the most critical flutter mode(s).

#### PROJECT DESIGN EFFORT

In contrast to the approach taken in the preliminary design phase, project design usually requires a more elaborate structural model than the simple Russell Beam or 2-D representation and consequently the use of large structural programs such as FAMAS or NASTRAN.

At this stage, the flutter characteristics of the design and the areas of the structure that are most amenable to correcting a flutter problem are well known. However, static reduction, equation (1), and further reduction of the dynamic equation to 50 degrees of freedom using natural vibration modes make the stiffness changes in equation (7) nonlinear in the design variables  $\beta_i$ , as well as a function of cross terms  $\beta_i \beta_j$ , etc. Because of large computing costs, the engineer attempts to extract the maximum design changes before either updating the structural vibration modes or updating the stiffness matrices. GFAM keeps the optimization process highly visible and permits the engineer to incorporate experience and judgement directly into the optimization process.



The updating of either vibration modes and/or the stiffness matrices requires the transfer back to the batch computing mode for the large order matrix computations.

## APPLICATIONS

GFAM applications, listed in Table 2, include flutter assessments for F-104 (Stores), L-1011, S-3A, NASA Arrow-Wing AST and preliminary design configurations. Table 2 also includes time history evaluations provided by ICSMP which are integrated in GFAM studies to include pilot flying characteristics requirements in synthesizing Control Configured Vehicle (CCV) systems.

The optimization for flutter performed during NASA AST Arrow-Wing study (Contract NAS1-12288) is an example of point design application of GFAM. Figure 23 defines the design regions for the outer section of the wing which were used to increase the flutter speed of the symmetric wing bending and torsion mode to 468 KEAS (see Figures 24 and 25). The wing sizing increments generated by GFAM's OPTX50TH program reflected sizing changes in the NASTRAN model. Because some of these sizings could not be translated directly into the physical model, the designers had to approximate the structure stiffness properties. The final NASTRAN model with the new sizings, however, produced a flutter speed of 512 KEAS (Figure 26). This indicates that further weight reduction is possible.

Some factors that determine whether GFAM will be used during a flutter analysis effort are

- 1) schedule constraints,
- 2) problem size,
- 3) analyst-in-the-loop requirements, and
- 4) batch cost comparison

GFAM computer runs are about 1.5 times the cost of the same task in batch. However, this may be compensated directly by showing that using GFAM will reduce the numerical effort by 35%. The dollar value of meeting schedule constraints is a variable that management determines on an individual basis.

## CONCLUDING REMARKS

Management responsible for flutter assessment and design modifications to satisfy flutter constraints views GFAM as an important computing system in fulfilling its charter in the design process. Successful GFAM users have been engineers with considerable interactive computing console time as well as engineers whose first interactive system exposure was GFAM. Cost, scope accessibility, and overall computer reliability are the primary factors affecting GFAM performance from the user viewpoint.

A true beneficiary of GFAM's architecture is methods development, which include Incremented Flutter Analysis (Reference 4), structural optimization

for flutter constraints (References 2 and 3), Flutter Modules Contract NAS1-12121 (Reference 1), Control Configured Vehicle - State Space model (Reference 5), and Control Configured Vehicle - Flutter model (IRAD effort).

GFAM is presently approaching the near-term goals defined by interactive flutter analysis requirements. Expansion to data preparation of large batch programs such as NASTRAN, aerodynamics programs, etc., and integration of the CADAM data base to go directly from loft data to flutter analysis with minimum data handling by the engineer, is presently under study.

#### ACKNOWLEDGEMENT

The author wishes to express his gratitude to Judy Skender, of Scientific Computing Division, for the successful development of GFAM's software, especially program MATRIX, and for many valuable discussions during the course of this work. Many thanks are also due Jim McIntosh, also of Scientific Computing Division, for his general assistance, in particular with the development of GFAM's control program.

#### REFERENCES

1. O'Connell, R. F., Hassig, H. J., and Radovcich, N. A.: Study of Flutter Related Computational Procedures for Minimum Weight Structural Sizing of Advanced Aircraft. NASA CR-2607, 1975.
2. Radovcich, N. A.: Structural Optimization with Flutter Speed and Minimum Gage Constraints. Report 26405, Lockheed-California Co., Apr. 1974.
3. O'Connell, R. F., Radovcich, N. A., and Hassig, H. J.: Structural Optimization with Flutter Speed Constraints Using Maximized Step Size. AIAA paper no. 75-778, 1975.
4. O'Connell, R. F.: Incremented Flutter Analysis. Journal of Aircraft, Vol. 11, No. 4, April 1974, pp. 236-240.
5. Davis, W. J., Chiodi, O. A., and Rabin, R. L.: Practical Application of Optimal Control Theory. AIAA paper no. 75-1030, Aug. 1975.

APPENDIX  
MATRIX LANGUAGE EXAMPLE

Code	Card Number
STRT	( 1)
PRNT DECK	( 2)
*	( 3)
* CCV Test Case Setup	( 4)
* 40th order problem reduced to 20th	( 5)
\$	( 6)
M @ 2644	( 7)
K @ 3002	( 8)
* VIB Analysis	( 9)
MI = INV(M)	(10)
A = MI * K	(11)
ROOT = EIG(A)	(12)
V = VECT(A,ROOT)	(13)
* Generate Matrices	(14)
S = -1.	(15)
UN = UNIT(1)	(16)
NROW = NULL(1,40)	(17)
NCOL = NULL(40,1)	(18)
NC20 = NULL(20,1)	(19)
NR20 = NULL(1,20)	(20)
NUN = S * UN	(21)
* DELTA	(22)
DELT = STOR UN AT (35,1) INTO NCOL	(23)
DELA = STOR NUN AT (36,1) INTO DELT	(24)
* MU	(25)
MU = STOR UN AT (1,19) INTO NROW	(26)
* GAM	(27)
GAM = STOR UN AT (20,1) INTO NC20	(28)
* LAM	(29)
LAM = STOR UN AT (1,19) INTO NROW	(30)

Code	Card Number
*	(31)
* GENERATE T MATRIX	(32)
ATR = STOR NR20 AT (39,1) INTO V	(33)
BT = STOR NCOL AT (1,20) INTO ATR	(34)
T = STOR UN AT (39,20) INTO BT	(35)
* GENERATE MODAL M,K,A	(36)
MM = T'*M*T	(37)
KM = T' * K * T	(38)
A @ 1864 TO 1873	(39)
SET (,9)	(40)
AM = T' * A * T	(41)
\$END	(42)
*	(43)
RNAM GAM AS 7999	(44)
RNAM LAM AS 8000	(45)
RNAM MM AS 8001	(46)
RNAM KM AS 8002	(47)
RNAM MU AS 8003	(48)
RNAM AM AS 8004	(49)
*	(50)
KEEP	(51)
*	(52)
PRNT ROOT,V,DELA,T	(53)
PRNT 7999 TO 8013	(54)

Card Number	Comments
( 1)	Initialize variable table
( 2)	Print this set of code
( 6)	Turn on automatic processing
( 7) - (8)	Set up variables M and K to represent matrices 2644 and 3002

Card Number	Comments
(10)	Compute inverse of M
(11)	Create matrix $A = M^{-1} * K$
(12)	Compute the eigenvalues of A
(13)	Compute the eigenvectors of A corresponding to the eigenvalues computed in card 12
(16)	Create a 1 by 1 unit matrix
(17) - (20)	Create matrices of zeroes
(21)	Create a 1 by 1 matrix with value -1
(23) - (24)	Create a 10 by 1 matrix, DELA, which is zero everywhere except $DELA(35,1) = 1$ and $DELA(36,1) = -1$ .
(26)	Create a 1 by 40 matrix, MU, which is zero everywhere except $MU(1,19) = 1$
(28)	Create a 20 x 1 matrix, GAM, which is zero everywhere except $GAM(20,1) = 1$
(30)	Create a 1 by 40 matrix, LAM, which is zero everywhere except $GAM(20,1) = 1$
(33) - (35)	Create a 40 by 40 matrix which is the eigenvalue matrix V modified as follows: $T(39,J) = 0$ for $j = 1,19$ $T(I,20) = 0$ for $I = 1,19$ and for $I = 21,40$ $T(39,20) = 1$
(37) - (38)	Create matrices MM and KM by performing pre- and post-multiply operations on the matrix T
(39)	Set up the multiple-entry variable A to represent matrices 1864 to 1873
(40)	Set the set number in the Input Display to card 9
(41)	Create the matrix AM by performing a pre- and post-multiply operation on matrix T.
(42)	Turn off the automatic processing
(44) - (49)	Reassign matrices GAM,LAM,MM,KM,MJ, and AM with matrix numbers 7999 to 8013.
(51)	Keep as permanent matrices all temporary matrices with matrix numbers in the range 1000 to 9999, i.e., matrices 7999 to 8013
(53) - (54)	Print matrices ROOT,V,DELA,T and 7999 to 8013

TABLE 1. GFAM HARDWARE AND CAPABILITY SUMMARY

Terminal Requirements:	Vector graphics scope, IBM 2250 or Vector General with NOVA for interface		
Computer Resources:	IBM 360-91, dedicated core, 126K bytes (31K single precision words), shared CPU with batch, CADAM, etc.		
Data Base:	Matrix data stored in FAMAS system Special card input list - card image data sets		
Problem size constraints:			
o Matrix module	Small order matrices	100 x 100	
	Large order matrices	300 x 300 (under development)	
o Flutter modules	50 x 50 complex flutter matrix		
Operational Data:			
o Wait times at scope	100 x 100 eigenvalue problem		90 seconds
	50 x 50 flutter matrix root extraction		15 seconds
	50 x 50 optimization matrix root extraction		20 seconds
	100 x 100 matrix inversion (real)		40 seconds

TABLE 2.- GFAM APPLICATIONS

MODELS	GFAM										ICSMP	
	Structure Model	Aerodynamics Model	Aero-Structure Interface	Guyan Reduction	Vibration	Modalization	Flutter	Structural Opt. For Flutter	CCV	Parametric Variations	Standard	CCV
F-104 Stores	20	X		NR	X	NR	20			X		
NASA Arrow-Wing							20/50	1		X		
ASW (S-3A)							50	2		X		
L-1011 - Full Size Model							20/50		X	X		
Preliminary Design Fighter	38			NR	X	NR	38	1		X		
L-1011 State Space Model					80				X			
L-1011 Model A	67		X	X	40	20	20/40			X		
L-1011 Demonstration Model	50			NR	X	NR	50	2		X		
L-1011 (Rigid) Equations of Motion Model	Number of structural elements				Problem order		Problem order	Number of simultaneous flutter constraints			X	X

NR: Not Required

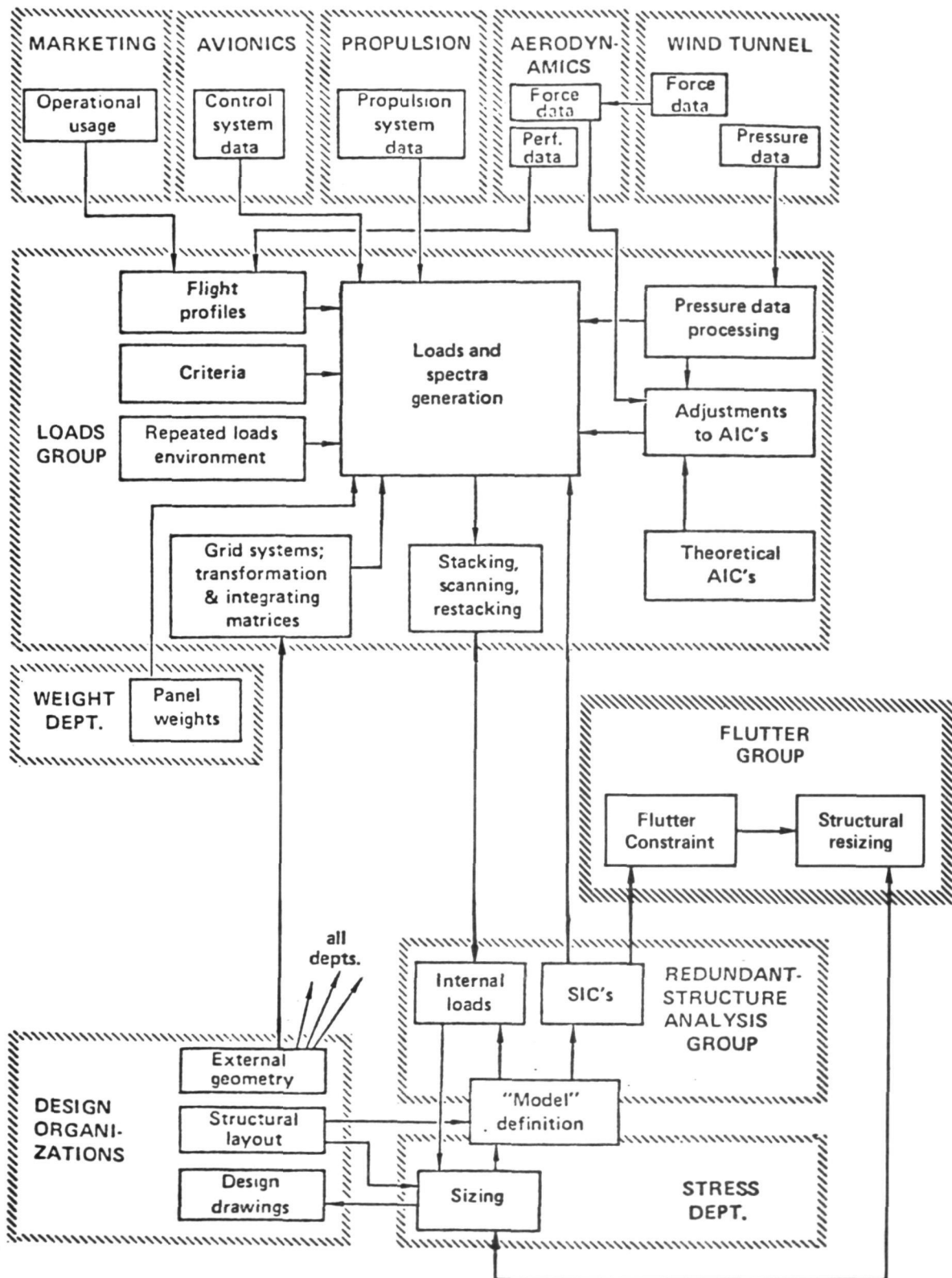


Figure 1.- Organizational interfaces and flow of data.



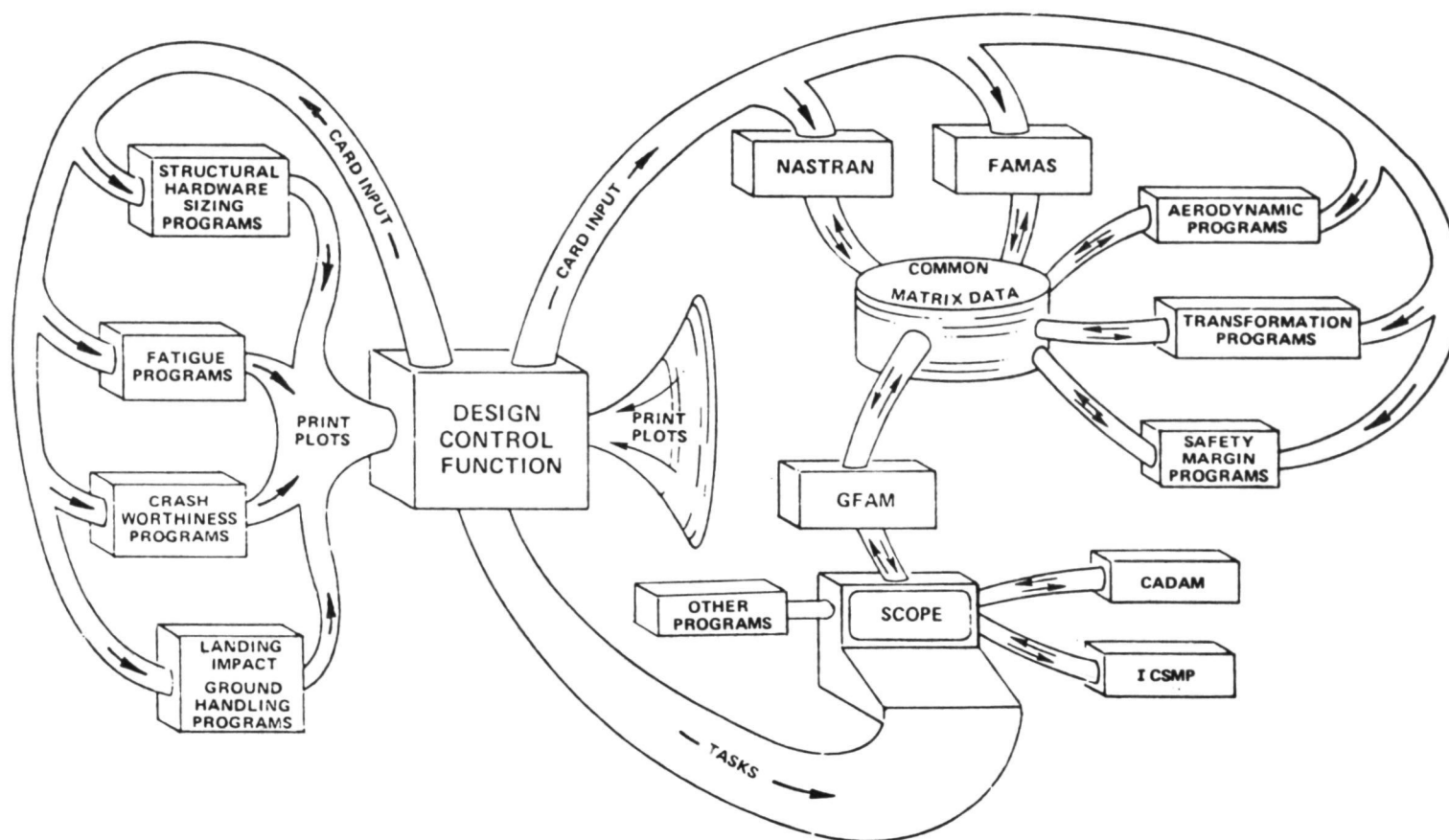


Figure 2.- Point design - Structures / Aeromechanics.

INTERACTIVE GRAPHICS COMPUTING SYSTEM - COMPATIBLE WITH FAMAS MATRIX DATA STORAGE SYSTEM

GFAM/MATRIX

- INPUT/OUTPUT/PRINT. FAMAS MATRIX DATA FROM TAPES, DISCS, AND CARDS. DATA MANAGEMENT SYSTEM
- PLOTTING/DATA SCREENING/TEST - THEORETICAL MATCHING
- MATRIX ALGEBRA. ADD/SUBTRACT, MULTIPLY, TRANSPOSE, INVERSE, EIGENVECTOR, EIGENVALUES
- MATRIX OPERATIONS. EXTRACT, STORE, UNIT AND NULL MATRIX GENERATORS.
- MATRIX ALGEBRA LANGUAGE.  $A = B + C$ ,  $A = B' * C * B$

$A = INV(B)...$

INTERACTIVE COMPILER/CHECKER/MATRIX  
DATA EDITOR

INLINE CODING, GO TO ....

GFAM/TECHNOLOGY  
MODULES

- FLUTTER - SOLVES THE FLUTTER EQUATION FOR DAMPING AND FREQ. TRACKS  
A GIVEN MODE. f-V-g PLOTS ON SCOPE
- FLUTTER VEL - COMPUTES FLUTTER VELOCITY DIRECTLY
- OPTX50TH - STRUCTURAL OPTIMIZATION FOR FLUTTER VELOCITY AND MINIMUM  
GAGE CONSTRAINTS
- FLUTTER FEED- COMPUTES FEEDBACK AMPLITUDE AND PHASE VS FREQUENCY  
REQUIRED FOR PRESCRIBED DAMPING VS VELOCITY
- BLOWOUT - COMPUTES PRESSURE TIME HISTORIES, AIRPLANE DECOMPRESSION  
ANALYSIS

Figure 3.- Graphics Flutter Analysis Methods,

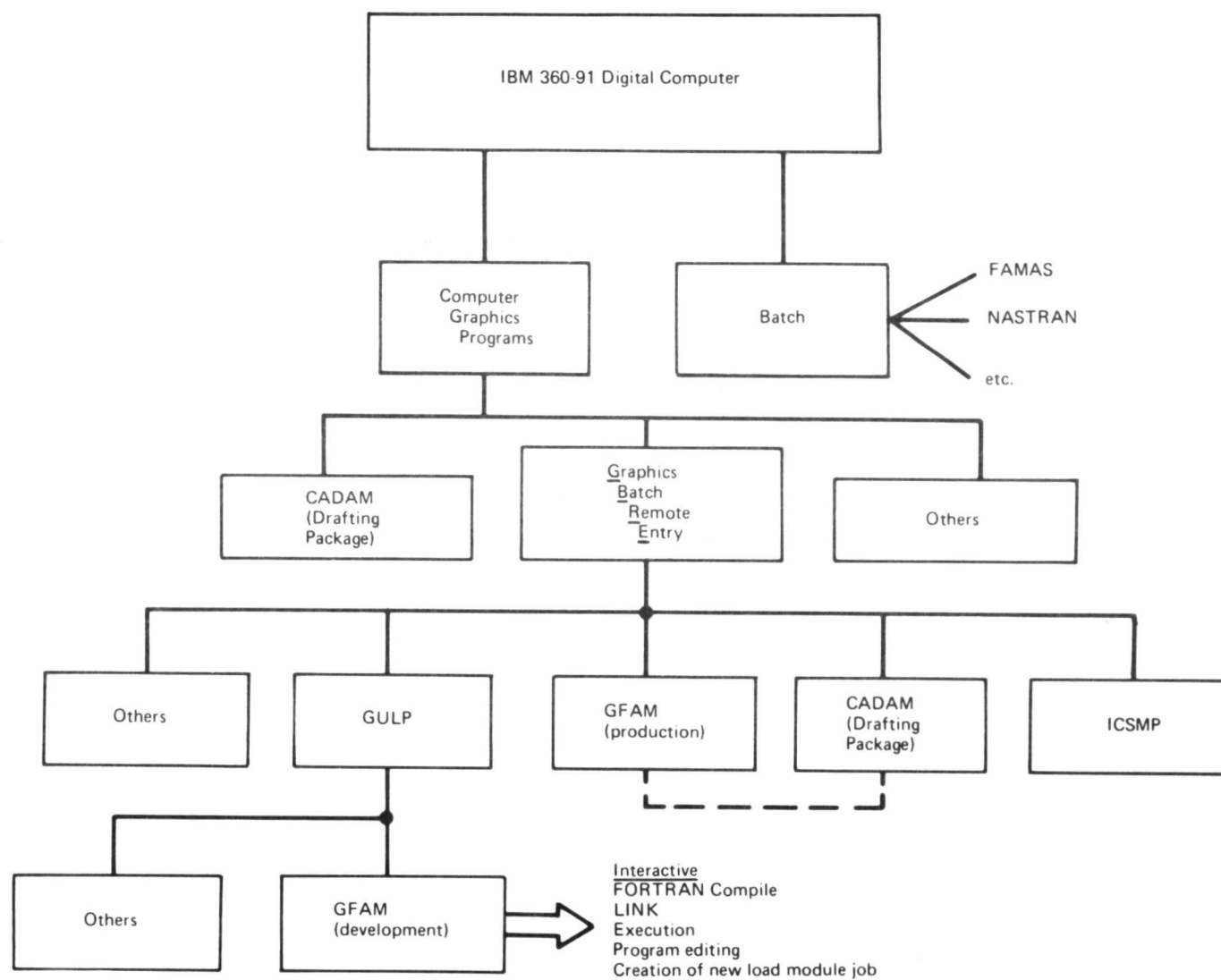


Figure 4.- GFAM in the computer tree.

GFAM CONTROL PROGRAM		THURSDAY	12/05/74
FT05F001	JCL-DATA	PROGRAMS	
ADAIS	FILE AG	ADAIS	
CCV	FILE BD	FLUTTER	
DATASET	FILE JD	FLUTTERP	
DAT11ST	FILE MD	FLUTFEED	
FATAST	FILE OC	FLUTTERM	
FATAST2	FILEDEM	F. VEL	
FAT1VAN	FILETEP	F. VEL Z	
FATPAR0		OPT 20TH	
FATRF0		OPTX50TH	
FATSET		OPTP50TH	
FATTEST		EDIT-OLD	
FLTVAG		MATRIX	
FLUTAG		MAT TEST	
FLUTBYD		I.F.A.	
FLUTEST		GRBEFAST	
FLUTJUR		PREADAIS	
FLUTNEW		INITDISC	
FLUTTER		BLOWOUT	
FLUT104		COMPLY	
FOLDEST			
F104S			
JUDYTES			
JURDAT			
NAR			
VIB			
FETCH      SAVE      PRINT      PURGE      SUBMIT      OUTPUT      LEAVE			

Figure 5.- GFAM control display.

12/04/74*****FLUTMAT USER TITLE *****FLUTMATJ VER 2					
INPUT DISPLAY					
PICK ONE OF THE MENU OPTIONS BELOW					
SET #	0	FREQUENCY	0.0	DAMPING	0.0
KEYWORD					
UNIT 1:	IPDS	256	TRX 0	RECRD 1	
UNIT 2:	IPDS	2163712	TRX 33	RECRD 4	KOUNT 27
/ CHKPNT / DATA READ / MATRIX TABLE DISPLAY / CARDS / PROGRAM / EXIT / / RESTART / DATA WRITE /					

Figure 6.- MATRIX- input display.







# OF MATRICES 27		DATA WRITE DISPLAY					
		LINE INCREMENT 5					
1	MATRIX	ROWS	COLS	TYPE	IFREQ	IDAMP	SET # FILE
1	1010	10	2	REAL	0	0	0 2
2	2644	40	40	REAL	0	0	15 2
3	1996	6	6	REAL	0	0	0 2
4	3002	40	40	REAL	0	0	0 2
5	1863	40	80	CPLX	7595	0	11 2
6	1864	40	80	CPLX	6295	0	11 2
7	1865	40	80	CPLX	5295	0	11 2
8	1866	40	80	CPLX	4395	0	11 2
9	1867	40	80	CPLX	3595	0	11 2
10	1868	40	80	CPLX	2995	0	11 2
11	1869	40	80	CPLX	2395	0	11 2
12	1870	40	80	CPLX	1995	0	11 2
13	1871	40	80	CPLX	1995	0	11 2
14	1872	40	80	CPLX	5	0	11 2
15	1873	40	80	CPLX	0	0	11 2

#FLAG MATRIX 0 TO MATRIX 0

#FLAG ALL MATRICES WITH SET 0

THE FOLLOWING IDENTIFIERS ARE REQUIRED FOR THE WRITE TAPE OPTION.  
IF LABEL # = 0 THEN THE MAT MATRIX WILL NOT BE PUT ON THE TAPE.

# JOB # 0 SEC # 0 LABEL # FOR MAT MATRIX 0

/ REWIND / MORE / SET FLAGS / ERASE FLAGS / FLAG ALL / RETURN /  
/ WRITE TAPE / WRITE DISC /

Figure 13.- MATRIX- data write display.

12/04/74*****FLUTMAT USER TITLE *****FLUTMATJ VER 2									
MAT TABLE DISPLAY									
COUNT 27	UNIT 1: IPOS	256	TRK 0	REC 1	MORE INC 15				
	UNIT 2: IPOS	2163712	TRK 33	REC 4					
1	MAT #	ROWS	COLS	TYPE	FREQ	DAMP	FILE	FILE POS	SET #
1 <>	1010	10	2	REAL	0.0	0.0	2	256	0
2 <>	2644	40	40	REAL	0.0	0.0	2	512	15
3 <>	1996	6	6	REAL	0.0	0.0	2	66048	0
4 <>	3002	40	40	REAL	0.0	0.0	2	66304	0
5 <>	1863	40	80	CPLX	0.760	0.0	2	131840	11
6 <>	1864	40	80	CPLX	0.630	0.0	2	243424	11
7 <>	1865	40	80	CPLX	0.530	0.0	2	395008	11
8 <>	1866	40	80	CPLX	0.440	0.0	2	540336	11
9 <>	1867	40	80	CPLX	0.360	0.0	2	721920	11
10 <>	1868	40	80	CPLX	0.300	0.0	2	853504	11
11 <>	1869	40	80	CPLX	0.240	0.0	2	985088	11
12 <>	1870	40	80	CPLX	0.200	0.0	2	1180416	11
13 <>	1871	40	80	CPLX	0.100	0.0	2	1312000	11
14 <>	1872	40	80	CPLX	0.001	0.0	2	1443584	11
15 <>	1873	40	80	CPLX	0.0	0.0	2	1638912	11

FREQ CHANGE STARTING AT 0.0 INCREMENTING BY 0.0  
FOR MATRICES 0 TO 0

DAMP CHANGE STARTING AT 0.0 INCREMENTING BY 0.0  
FOR MATRICES 0 TO 0

SET # CHANGE STARTING AT 0 INCREMENTING BY 0  
FOR MATRICES 0 TO 0

SET PURGE FLAGS ON MATRICES 0 TO 0 OR FOR SET # 0

/ RESTORE / REWIND / MORE / SET PARAMS / PURGE / REBUILD / RETURN /

Figure 14.- MATRIX- matrix table display.



SYMMETRIC FLUTTER ANALYSIS - CHORDWISE STIFFENED ARRANGEMENT  
MACH NO. = 0.6  
WEIGHT = 321,000 LBS

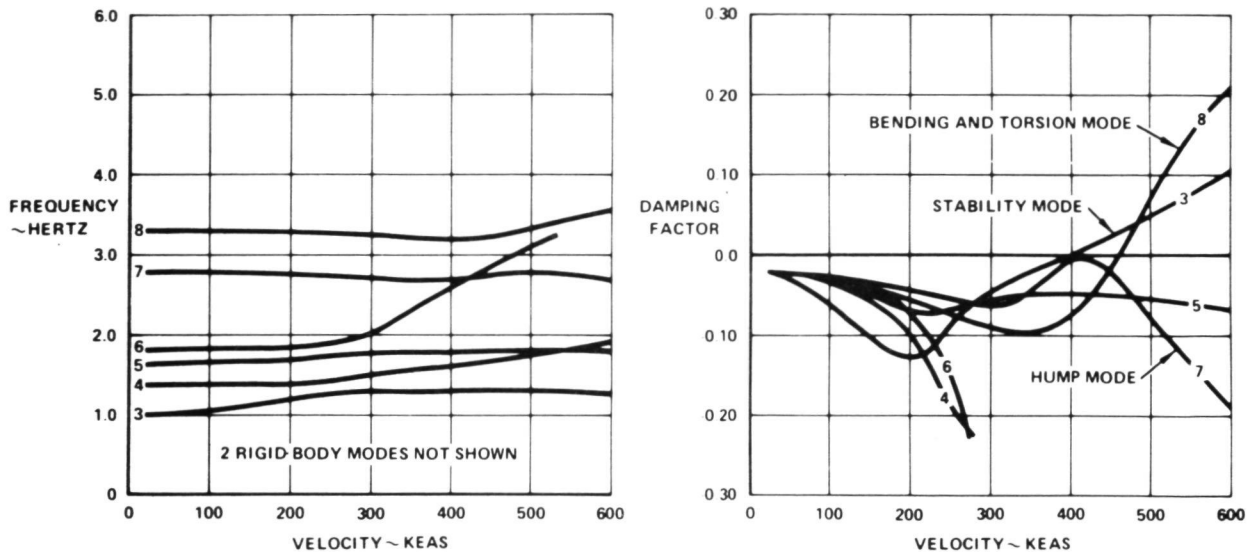


Figure 15.- Typical flutter f-V-g plots. 321 000 lb = 1427.8 kN.

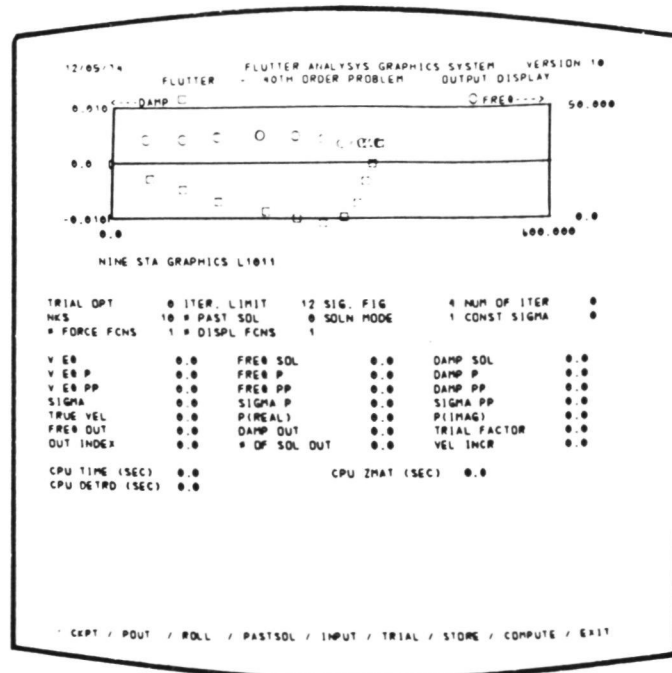


Figure 16.- FLUTTER- output display.

```

12/05/74          FLUTTER ANALYSIS GRAPHICS SYSTEM  VERSION 10
                   FLUTTER - 40TH ORDER PROBLEM      INPUT DISPLAY

NINE STA GRAPHICS L1011

FORCE T.F. 1 8900 FORCE T.F. 2 0 FORCE T.F. 3 0 FORCE T.F. 4 0
FORCE T.F. 5 0 FORCE T.F. 6 0 FORCE T.F. 7 0 FORCE T.F. 8 0
FORCE T.F. 9 0 FORCE T.F. 10 0 FORCE T.F. 11 0 FORCE T.F. 12 0
DISPL T.F. 1 8901 DISPL T.F. 2 0 DISPL T.F. 3 0 DISPL T.F. 4 0
DISPL T.F. 5 0 DISPL T.F. 6 0 DISPL T.F. 7 0 DISPL T.F. 8 0
DISPL T.F. 9 0 DISPL T.F. 10 0 DISPL T.F. 11 0 DISPL T.F. 12 0
MODIFY F TF 0 MODIFY D TF 0 KEEP F TF 0 AS 0
KEEP D TF 0 AS 0

T MATRIX NUM. 8885 WEIGHT MAT. NUM. 8886 STIFF MAT NUM. 8887
AERO SET NUM. 91 P-AERO OPTION 0 DAMP MATRIX NUM. 0
HORN DELK 0

STR. DAMP. 0.0200 REF. LENGTH 269.8999 MACH NUM. 0.8800
DEG OVER STD 0.0 AERO SCALE 1.0000 SPEED CONV. 20.2537
GRAY CONST 1.0000 MUNT OPT 0.0 K SCALAR 0.0

STD AIR DENSITY 0.1146262881096-06

MODIFY / KEEP / RESTART / READ / OUTPUT / START / EXIT /

```

Figure 17.- FLUTFEED- input display.

```

FORCE TF 1
DELTA MATRIX NO. 8881

SET 1 OF 1
MU MATRIX NO. 8882

TF 1 OF 4 TYPE 0
NUM .15230 E 08 .30460 E 08 .15230 E 08
DENOM .40000 E 03 .40000 E 02 .10000 E 01
TF 2 TYPE 0
NUM .00000 .72000 .10000 E 01
DENOM .28900 E 03 .34100 E 02 .00000
TF 3 TYPE 0
NUM .60000 E 01 .10000 E 01 .00000
DENOM .19610 E 02 .10000 E 01 .00000
TF 4 TYPE 0
NUM .00000 .00000 .00000
DENOM .10000 E 01 .53830 E 01 .10820

ADD TF / DELETE TF / ADD SET / DELETE SET / UP / DOWN / EXPLAIN TYPE /
MODIFY DELTA / MODIFY MU / KEEP DELTA / KEEP MU / RETURN /

```

Figure 18.- FLUTFEED- transfer function display.



Figure 19.- FLUTFEED- matrix modify display.

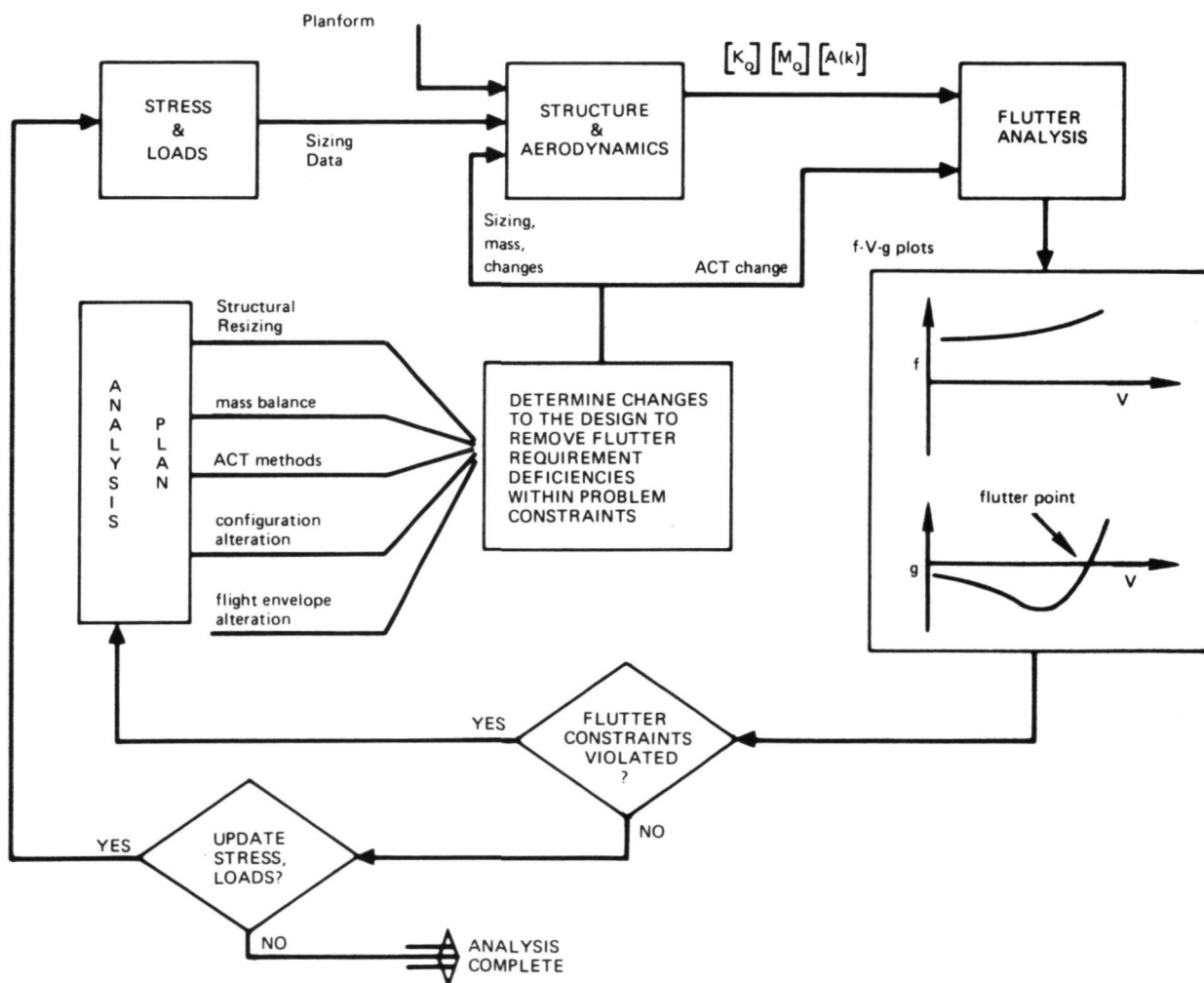


Figure 20.- Vehicle design cycle for flutter, stress, and loads,



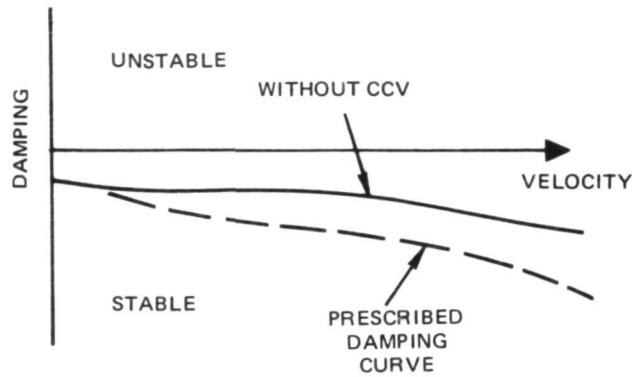


Figure 22.- Prescribed damping versus velocity,

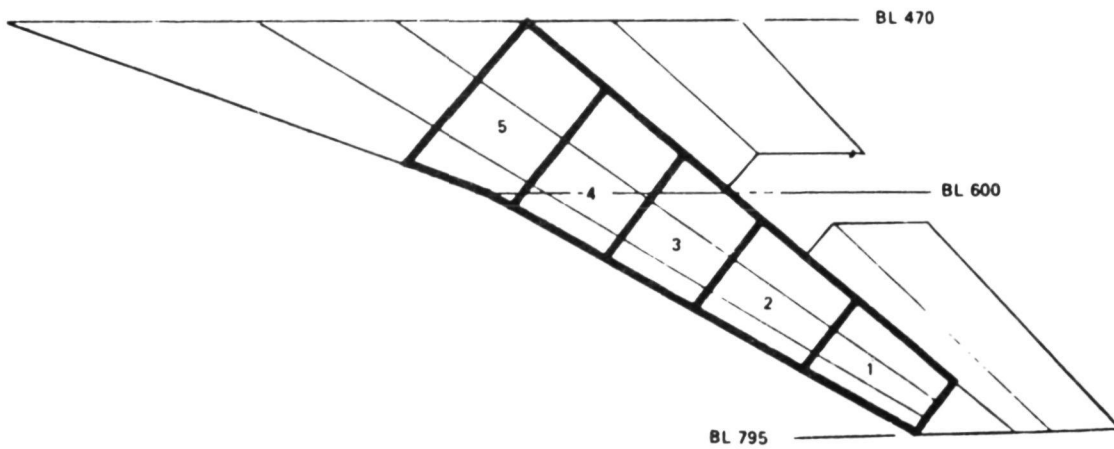


Figure 23.- Flutter optimization design regions,

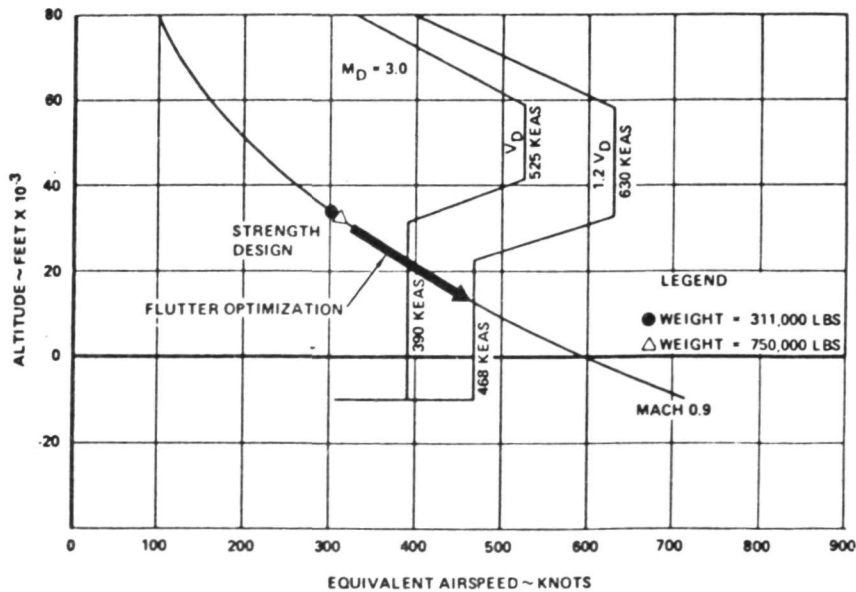


Figure 24.- Flutter speeds for symmetric bending and torsion mode.  
 311 000 lb = 1383.3 kN; 750 000 lb = 3336 kN.

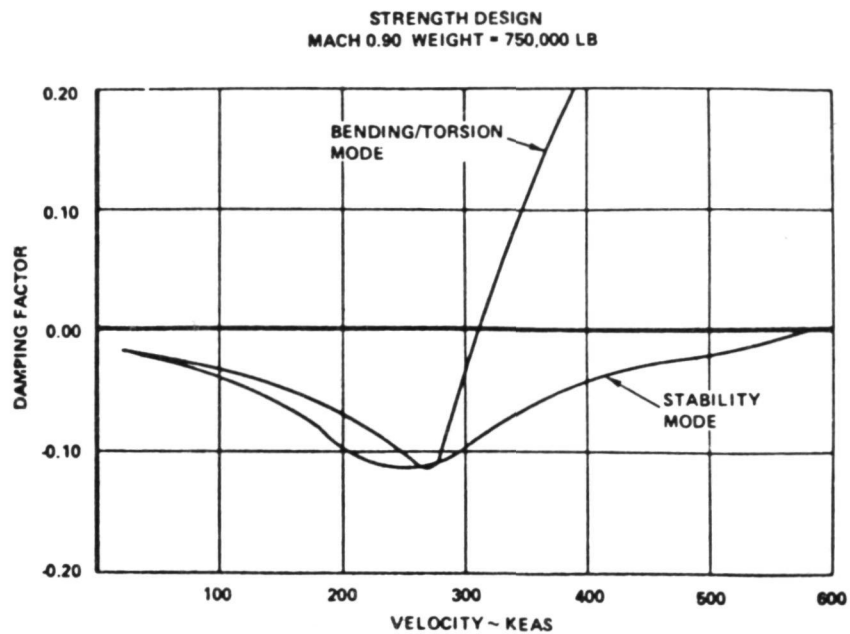


Figure 25.- Symmetric flutter analysis - Mach 0.9 - FFFP. KEAS denotes knots equivalent airspeed. 750 000 lb = 3336 kN.

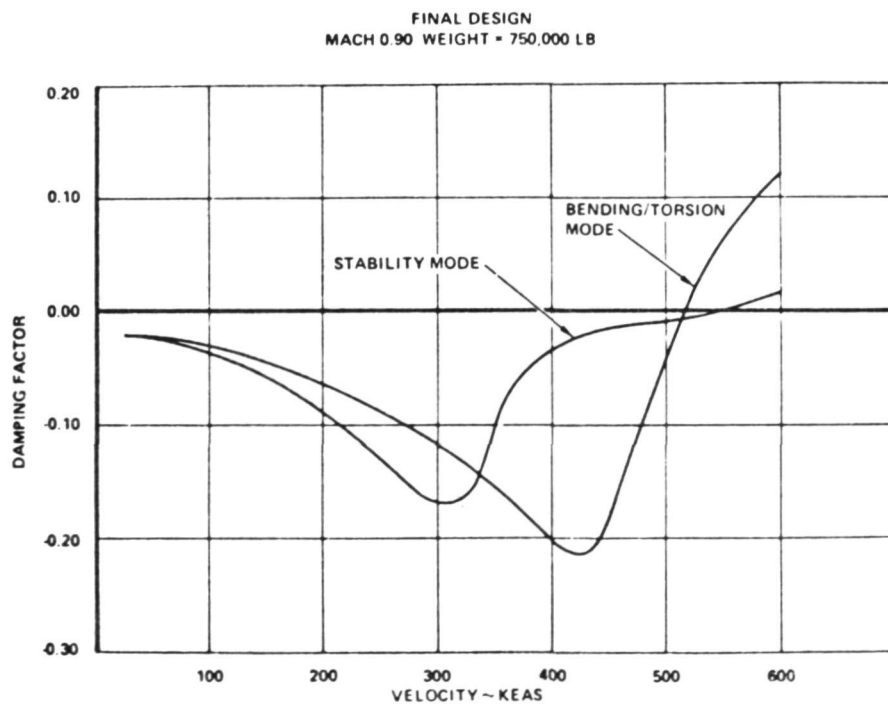


Figure 26.- Symmetric flutter analysis - Mach 0.9 - FFFP. KEAS denotes knots equivalent airspeed. 750 000 lb = 3336 kN.

**Page Intentionally Left Blank**

## INTERACTIVE COMPUTER AIDED TECHNOLOGY: EVOLUTION IN THE DESIGN/MANUFACTURING PROCESS

C.H. English  
Section Manager - Design  
Computer Aided Technology Project  
McDonnell Aircraft Company  
P.O. Box 516  
St. Louis, Missouri 63166

### ABSTRACT

A description is given of a powerful computer-operated three dimensional graphic system and associated auxiliary computer equipment used in advanced design, production design, and manufacturing at McDonnell Aircraft Company in St. Louis, Missouri. This system has made these activities more productive than when using older and more conventional methods to design and build aerospace vehicles.

The process of designing and manufacturing the complex parts of the aerospace industry requires the coordinated efforts of many technical disciplines. Effective interface among these groups demands that each understand the design engineer's concept completely. Heretofore, the only medium for communicating the concept has been the two dimensional engineering drawing; however, it was difficult to ensure that every discipline was working with the latest revision to the drawing and that every individual interpreted the drawing the same way. This leads to the possibility of erroneous drawing interpretations, mis-machined hardware, and many pieces of corrective action paperwork, all contributing needless increased costs and program schedule delays.

Today, designers are no longer restricted to creating part, or entire vehicle definitions within the constraints of 2-dimensional descriptive geometry. With the use of this graphic system, they are now able to define parts using a wide variety of geometric entities, define parts as fully surfaced 3-dimensional models as well as "wire-frame" models. Once geometrically defined, the designer is able to take section cuts of the surfaced model and automatically determine all of the section properties of the planar cut, lightpen detect all of the surface patches and automatically determine the volume and weight of the part. He may also view the model from any vantage point desired by

rotating it about any designated spatial axis, thus providing better visibility of its geometric definition. Further, his designs are defined mathematically at a degree of accuracy never before achievable. The mathematically defined model, stored in the central computer, is accessible to other engineering disciplines, tool designers, loft, manufacturing planning, quality assurance, and other personnel having a need for these data. This provides them with a single source set of geometric data with which they are able to address the tasks of analyzing the designs for performance, structural integrity, fit and function, part fabrication, and inspection of parts after fabrication. These tasks are accomplished using various computer aided techniques, software modules, and computer hardware, each tailored to do the job for which it is best suited.

The use of this computerized system has proved to be both cost and time effective in the conduct of our business. With continued use and development, even greater cost and time saving techniques are anticipated.

### INTRODUCTION

In today's very competitive business world, each business firm must continually seek improved methods of conducting its varied activities if it is to survive and retain its competitive position. These improved methods are normally measured in terms of reduced costs, reduced manhours, reduced lead-time, better products, and maintenance of a creative work environment.

The design of an aerospace vehicle is predicated on a multitude of requirements of each of the systems and technical disciplines comprising the total design activity. In order that each technical discipline be properly interfaced with all other disciplines, and the integrity of the vehicle performance and specification requirements maintained, close intergroup coordination is mandatory.



During any design program, several design interactions are required as a result of loads analysis, weight and balance requirements, aerodynamic considerations, and other design ramifications. As a result of these necessary design interactions, there is often difficulty in effecting a smooth and orderly response to the required alignment of the specific design tasks to satisfy these requirements. Reaction time by all affected parties must be closely coordinated to prevent jeopardizing major program milestone schedules. Design changes vary from being minor in nature to the very complex, but all require that we exploit every available technique, talent, and design tool at our disposal to assure that the changes are made on a timely basis to prevent schedule compromises. Computer Aided Technology, which is comprised of numerous types of computer hardware and software, and used by our engineering and manufacturing disciplines at MCAIR, is helping us to meet these goals.

MCAIR has formed a Computer Aided Technology (CAT) project, made up of representatives from departments that use computers in their work, for the purpose of providing the coordination necessary to effect smooth and cost effective interfaces between each of the activities and to identify and eliminate duplication of efforts in the development of software. Our CAT project activities are similar to those of other companies currently developing interactive computer graphic capabilities, but is probably structured differently. Our organizational structure is shown in Fig. 1.

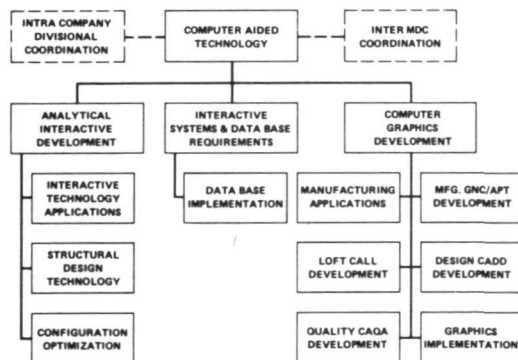


Fig. 1 CAT Organization Chart

Our computer system has been modularized into several specific packages in order that each department using computer graphics be required to develop and maintain only the software necessary to accomplish its required departmental tasks. Training of the console operators for each module is therefore simplified compared with having to instruct the operators how to use a single large module that would be used by all departments.

Our CAT project personnel are co-located, providing close coordination and a direct line of communication between the various disciplines in the development of their modules.

There are currently six basic graphic modules under development or coordination by the CAT project. They are Computer Aided Design-Drafting (CADD), Interactive Computer Aided Design Evaluation (ICADE), Computer Aided Loft Lines (CALL), Computer Graphics Structural Analysis (CGSA), Graphic Numerical Control (GNC), and Computer Aided Quality Assurance (CAQA). Their interrelations are depicted in Fig. 2. Each of these modules is discussed in the following paragraphs.

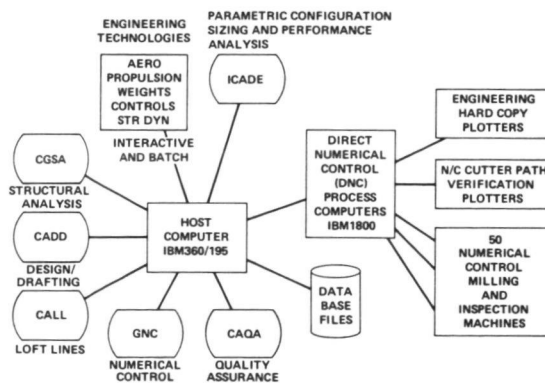


Fig. 2 McDonnell Aircraft Computer Aided Technology

### Computer Aided Design-Drafting (CADD)

The term CADD (pronounced caddy) denotes various computer techniques and applications where data are either presented or accepted by a computer in a geometric form as opposed to alpha-numerics only. CADD is "interactive", which implies that there is an efficient, real time interplay of actions between the console operator and

the system hardware devices. CADD therefore describes an interactive and conversational mode of operation, utilizing a display console where the engineer may describe his design, perform analysis procedures, and make changes to the design if he so chooses.

A designer is normally concerned with creating a geometric representation of a physical object. In the drawing board mode, these are lines drawn on paper or mylar in two dimensions. This representation, when the CADD package is used, is in terms of an exact mathematical description in the computer's memory, either as a two dimensional lines drawing, a three-dimensional wireframe drawing, or as a completely surfaced definition of the model in three dimensions. In addition, the graphic representation is displayed on the cathode ray tube (CRT) (Fig. 3). Hardware and software features enable the operator to converse with the computer by detecting, with a light pen, elements displayed on the CRT screen and by inputting specific instructions with the alphanumeric and functional keyboards.



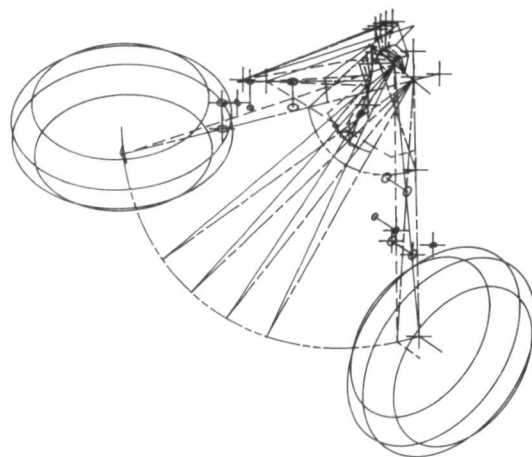
**Fig. 3 Model Graphically Displayed**

The CADD system places the designer in command of the computer while he retains the essence of his normal working environment. As a result, he is able to address and solve problems in a geometric language with which he is familiar and comfortable rather than having to understand the intricacies of the computer or programming languages.

The CADD system enables the engineer and computer to function as a problem-solving team. The significance of the team effort is that each member is free to do the part of a task for which

he is best suited. The computer can perform the complex geometrical tasks, or the repetitive tasks, while the engineer can focus all of his creative talents to the problem rationale.

A CADD console was first used in a production environment for direct support of the F-15 project in July 1970. It was initially utilized for the purpose of synthesizing the three-dimensional spatial geometry and path of travel of the main landing gear mechanism (Fig. 4) to assure that the required precision of motion was obtained. It was later used to define many of the components of the fuselage and related structure (Fig. 5).



**Fig. 4 F-15 Landing Gear Kinematics**



**Fig. 5 F-15 Fuselage Structural Component**

Although some of the more general applications at MCAIR have been layouts and detail design of structural components and complex mechanisms not associated with the aircraft moldline, the ability to access lofted surface definitions of various aircraft via the CRT has been important to the designer. With lofted surface definitions available to the operator, he is capable of generating any section cut on the airplane to assist him in determining the best design approach for the configuration of a system component. Once the design approach is determined, the operator is able to offset from moldline a distance equal to the skin thickness and have the entire periphery of the structural component completely defined. The internal portion of the structure, including the flanges, pockets, and stiffeners are added in a 3-D wireframe format. The wireframe model is then converted to a surfaced model with each surface identified by a label. Labels are used as identifiers for accessing the geometric data of the surfaces from the computer.

CADD has also been used on several of our Advanced Design programs for defining vehicle configurations, landing gear mechanism geometry, external stores arrangements, and various vehicle system tradeoff studies.

Whenever a hardcopy drawing of the data displayed on the CRT is desired, the operator merely inputs pertinent drawing information into the computer, such as, scale, drawing size, type of drawing material, and any special instructions desired by the operator. These computer stored data are then queued for hardcopy, utilizing a computer driven automated plotter (Fig. 6) and, overnight, the completed hardcopy is delivered to the requestor.

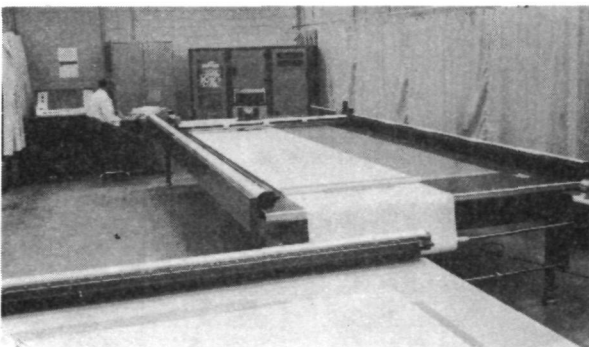


Fig. 6 Automated Plotter

Many disciplines are currently interfacing the CADD module, each accessing the single source geometric data created by the designer. They include Strength, Propulsion, Aerodynamics, Loads, Weights, Structural Dynamics, Thermodynamics, Operations Analysis, Wind Tunnel Design, Structural Dynamics Labs, Loft, Tool Design, Master Layout, Manufacturing Planning, and Quality Assurance. Each are accomplishing their required tasks in a more efficient and time saving manner than in the past as a result of this interface.

#### **Interactive Computer Aided Design Evaluation (ICADE)**

When an aircraft configuration has been synthesized on the CRT and stored in the computer, members of the design team responsible for performance analysis can access geometric data at the same console used to create the configuration, and apply an interactive parametric sizing analysis program called ICADE (pronounced eye'-kay-dee), to determine if the flight performance requirements and vehicle size are compatible. He also has the capability of sizing the aircraft configuration to satisfy a specified set of flight performance requirements, i.e., size wing, engine, and/or fuel volumes to meet or exceed performance requirements and determine concomitant changes in the aircraft geometry, mass properties, aerodynamic characteristics, installed engine performance, and life cycle cost characteristics. The system is quite effective when used to conduct studies for determining the effect of parametric variations in aircraft design parameters, (e.g., wing geometry, wing loading, thrust to weight ratio, structural load factors, etc.) on the aircraft size while holding the performance constant, or determining the effect of parametric changes in performance parameters (e.g., mission radius, specific excess power, etc.) on aircraft size. In addition, ICADE can be used to evaluate specific design or performance tradeoffs; determine the sensitivity of aircraft size to incremental changes in engine size, fuel quantity, fixed weight, aerodynamic drag, specific fuel consumption, etc., while holding the performance constant; and generate data required for the Parametric Design Analysis Procedure, a separate batch analysis program, which is used to determine "optimum" design parameter values which maximize or minimize a measure of aircraft effectiveness (e.g., takeoff

gross weight, life cycle cost, etc.) Figure 7 depicts the data flow of the program.

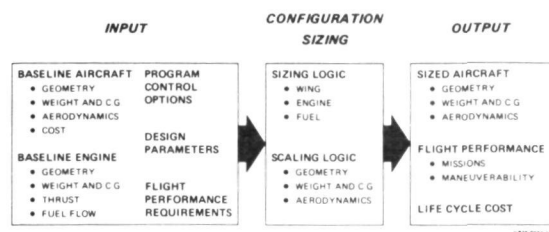


Fig. 7 ICADE Program Data Flow

All input data required by ICADE have been divided into thirteen categories to facilitate the editing process. These data categories are:

1. Design parameters
2. Program control indicators
3. Sensitivity parameters
4. Mission performance parameters
5. Maneuver performance parameters
6. Miscellaneous performance parameters
7. Geometric characteristics - baseline aircraft
8. Weight characteristics - baseline aircraft
9. CG characteristics - baseline aircraft
10. Aerodynamic characteristics - baseline aircraft
11. Propulsion system characteristics - reference engine
12. Cost characteristics - baseline aircraft
13. Payload characteristics

The analyst initiates the editing process by detecting with the light pen the data category to be edited from a menu display. The data from the selected category are then displayed in card image form identical to that used for the batch version of the program. A menu of commands is also displayed on the CRT.

During execution of the ICADE sizing iteration, the iteration number and current values of fundamental aircraft sizing parameters are displayed. These data provide the analyst with information on the convergence behavior of the sizing iteration, but do not allow him to interact with this phase of program execution.

After the sizing iteration has converged to a solution, and the cost and mission performance characteristics have been calculated, the following summary of the sized aircraft characteristics is displayed:

1. Geometric characteristics (comparison with baseline aircraft)
2. Weight characteristics (comparison with baseline aircraft)
3. CG characteristics (optional)
4. Mission performance
5. Maneuver performance
6. Cost characteristics (optional)
7. Aerodynamic characteristics

### Computer Aided Loft Lines (CALL)

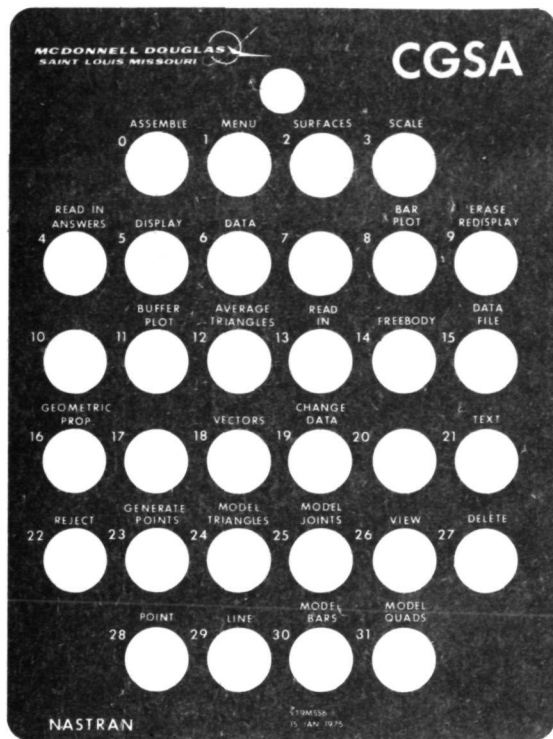
CALL is a module developed and utilized for the primary purpose of creating, modifying, and evaluating various surface geometry shapes interactively. Surface related geometry operations such as digitized data curve matching, continuous surface blending, creation of the CADD module entity types, and CADD drawing file system compatibility have been incorporated to provide the preliminary designer the ability to graphically define a surfaced vehicle in a format that is usable by other disciplines. Standard loft surface shapes, i.e., conic, ruled, parametric cubic, and general parametrics provide designers latitude in defining vehicle configurations and are helpful in determining fuel and equipment compartment volumes.

### Computer Graphics Structural Analysis (CGSA)

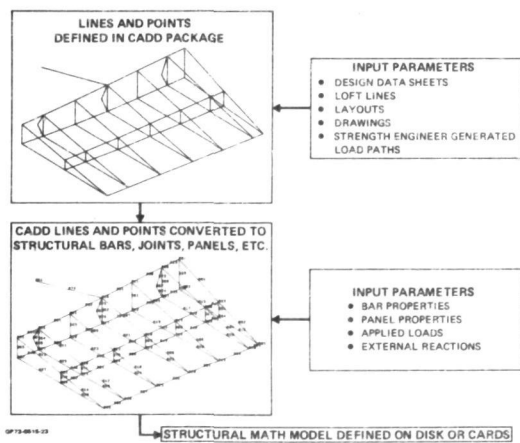
CGSA is a module that has been developed to aid structural analysis and design. CGSA is used to provide internal loads, stresses, and strains for structural sizing, and documentation of structural integrity. This system functions in conjunction with the CADD system.

A special overlay is used on the function keyboard (Fig. 8) when the CGSA module is in operation. The function keyboard is the same as the one used for all of our modules, but the function of each button has been software-altered for each module.

CGSA is used to prepare three dimensional finite element model input data for either the CASD or NASTRAN structural analysis programs (Fig. 9). (CASD, an acronym for Computer Aided Structural Design, is a structural analysis program developed by Douglas Aircraft Company prior to the development of NASTRAN by NASA).

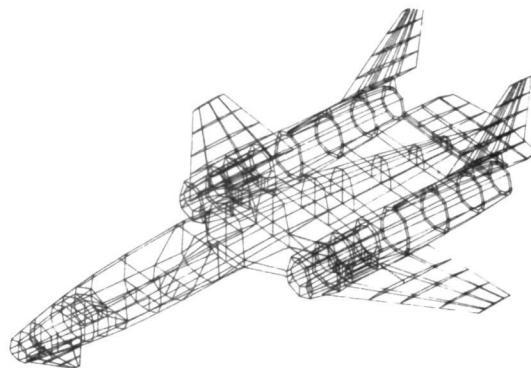


**Fig. 8 CGSA Function Keyboard Overlay**



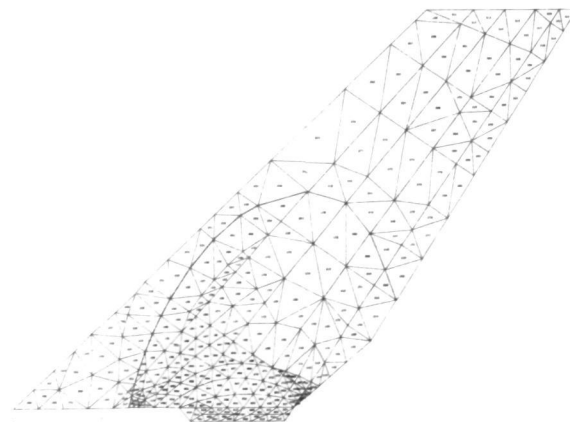
**Fig. 9 Structural Idealization**

Basic geometry is developed from loft data, conventional engineering drawings, or the three dimensional geometry of the vehicle constructed and filed in the computer by the designer (Fig. 10).

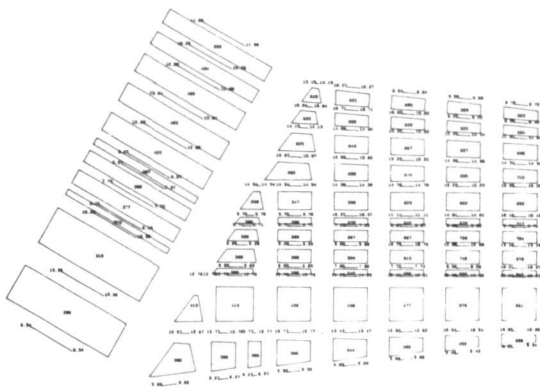


**Fig. 10 Basic Structural Geometry in Point Line Format**

This results in a point-line model which is converted to a finite element model by defining bar/panel/membrane element properties, applied loads, and external boundary conditions. The model may be viewed, checked, and edited at the console for verification of the input data. Hardcopies are produced displaying input properties and element labels (Fig. 11 and 12) and the model is then stored in the CADD drawing files for future retrieval. The CASD or NASTRAN input card images are automatically created on a disk data set for submittal to the structural analysis program. The utilization of CGSA for model generation eliminates the need for manually creating the models, input sheets, and punched cards.

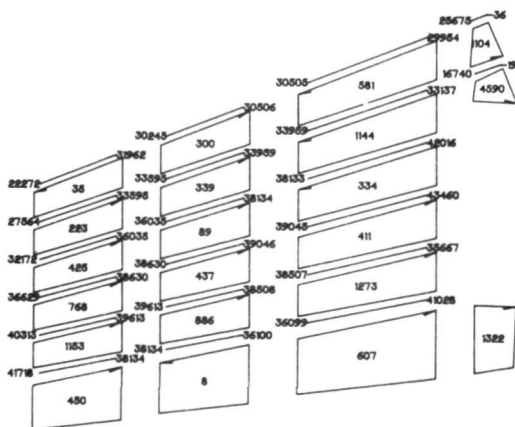


**Fig. 11 Triangular Membrane Labels**

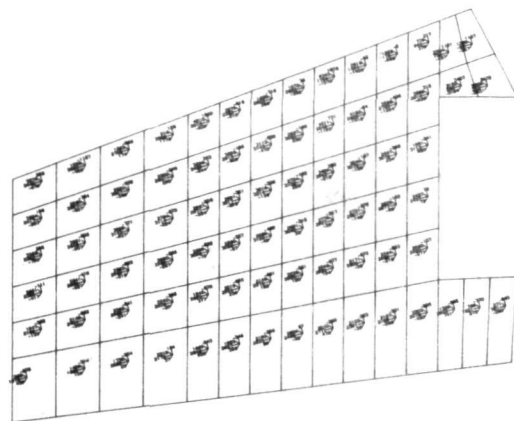


**Fig. 12 Upper Cover Bar Areas and Panel Thicknesses**

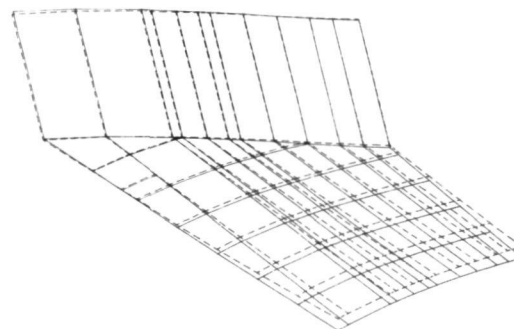
Upon completion of the CASD or NASTRAN structural analysis program (which performs a minimum strain energy solution), the results are stored with the model in the CADD drawing file and are also output in the more usual manner on an alphanumeric printer. The answers may be displayed at the console and hardcopied for all structural elements. The display options include magnitudes of bar load, bar stress, panel shear flow, panel shear stress, membrane stress, membrane strain, and membrane running loads, examples of which are shown in Fig. 13 and 14. Also the plotting of bar loads or stress, bending moments, and deflected shapes (Fig. 15) are available.



**Fig. 13 Bar Stress and Panel Shear Flow**



**Fig. 14 Skin Strains - Micro-in./in.**



**Fig. 15 DC-10 Upper Cover Deflected Shape**

The displays and hardcopies in the form of load sheets are used by engineering supervision to gain an overall feel for the structural analysis, and also by the detail stress analyst as inputs for his computations. This type of graphical presentation eliminates the need to manually prepare load sheets. This was previously done by reading the printed output data and associating it with the pictures of the structural idealization.

CGSA is also used to transmit, within the computer, structural flexibility influence coefficients or a reduced stiffness matrix to the Loads and Structural Dynamics Departments. Additional geometric data is passed to the Loads Department who then generate panel point loads which are transmitted back to the CGSA module.

CGSA is effective when used in both Advanced Design and Project environments. Advanced Design requires smaller models, quick turnaround,



and good visibility, and it adds to the technical credibility of technical proposals. In a Project environment, it is effective in handling large quantities of data (the F-15 Wing model has 10,000 card images) and in providing Design and Strength department personnel with essential data when responding to design changes and updates.

### Graphic Numerical Control (GNC)

When work is scheduled to start on the manufacturing cycle of a part, a GNC programmer is assigned the task of creating an Automatically Programmed Tool (APT) program for machining the part. The programmer retrieves a copy of the engineering drawing of the part from the computer files and within seconds, the configuration appears on the CRT screen as shown in Fig. 16. This may be the first time he has seen the drawing; consequently, the dimensions and mathematical properties of the part are unknown to him. The part must be evaluated sectionally, and to achieve visual simplification, he will probably isolate one pocket at a time to work on. He may do this by typing in the label numbers of each surface comprising the pocket. The computer searches its files and displays the pocket on the CRT.

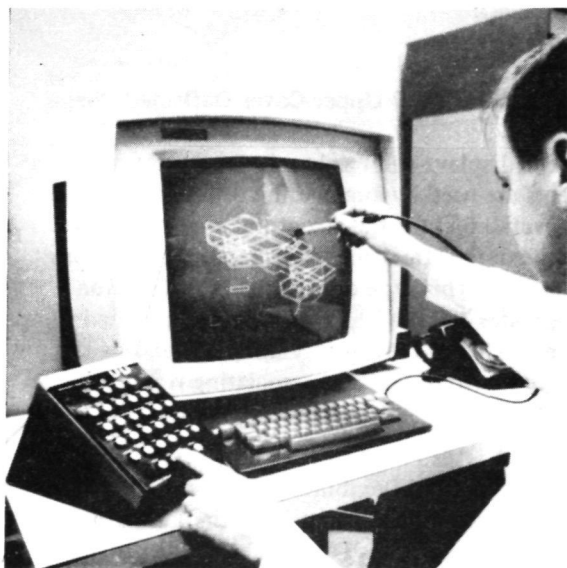


Fig. 16 Copy of Drawing Retrieved from Drawing Files

By depressing a specific function key, the programmer activates a routine which analyzes the connectivity of the part geometry, and extracts the geometric properties or mathematical description of the part and stores the data in the computer memory.

Another function key is then depressed and the programmer may then create the APT source statements. The setpoint of a cutting tool can be keyed-in using the alphanumeric keyboard, or can be lightpen detected. After the cutter setpoint location is established and accepted, various parameters necessary for defining the cutting operation are input: diameter of the cutter(s), spindle speed(s), check and drive surface(s), various tool axes, etc.

The APT program, created from the previous inputs, is then reviewed (Fig. 17). Using Data Set Edit, he can add, change, or delete APT statement from his program using the lightpen and alphanumeric keyboard, or by using a low cost VM 7000 terminal (Fig. 18). During the programmer's detects and inputs, APT source statements are simultaneously being generated.

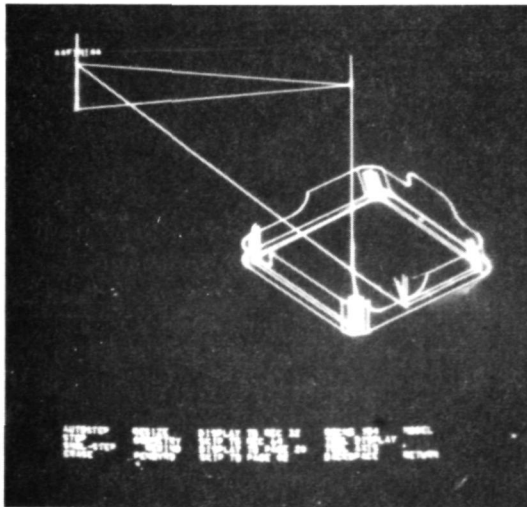


Fig. 17 Review of APT Source Program



**Fig. 18 A Low-Cost VM7000 Can Be Used for Data Set Edit**

A visual inspection of a cutting simulation then takes place. The programmer observes the motion of the cutter generated by the parts program. The path of travel of the center of the cutter is plotted and superimposed on the display of the part. The tool axes, tool end, and a representation of the tool itself can also be displayed to ensure that the parts program is correct (Fig. 19 and 20).



**Fig. 19 Displaying the Path of the Center of the Cutter as Well as the Tool Axes**



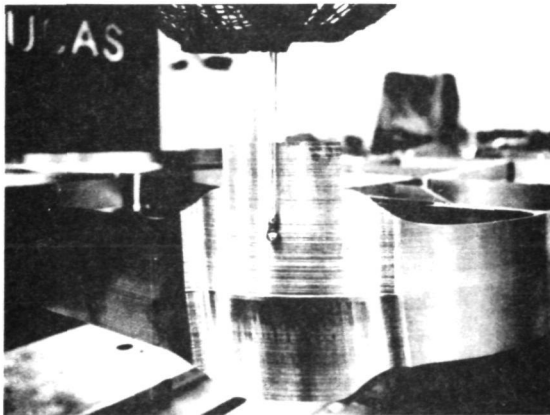
**Fig. 20 Displaying the Tool at a Critical Position**

If necessary, Data Set Edit is again used to correct errors. The program is then implemented to generate the machine tool data instructions for the actual machining of the part.

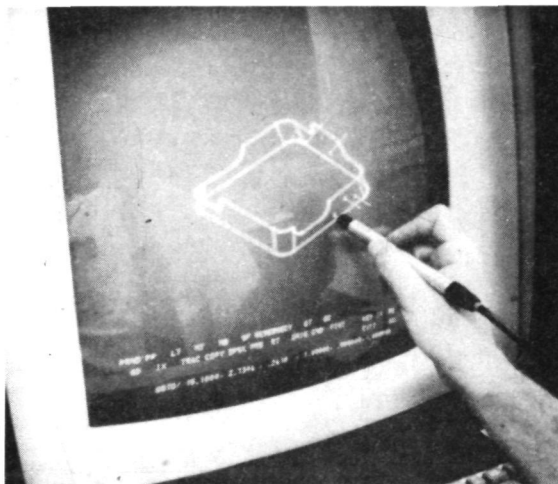
#### **Computer Aided Quality Assurance (CAQA)**

Upon completion of the machining operation, inspection of the part is necessary to determine if it accurately represents the design definition. A copy of the part definition that was defined by the designer is retrieved by the quality assurance programmer. Using the lightpen, he generates points on the surface of the part that the DNC inspection machine is to probe (Fig. 21). Vectors are displayed (Fig. 22) on the part at the points at which the probe is to make contact, and APT statements, which will guide the probe to these points on the actual part, are created as the vectors appear. He next reviews the program, using Data Set Edit if required, and then, as in the case of the centerline plot, it is visually verified (Fig. 23), edited if required, and finally sent to an IBM 1800 computer for the inspection operation.

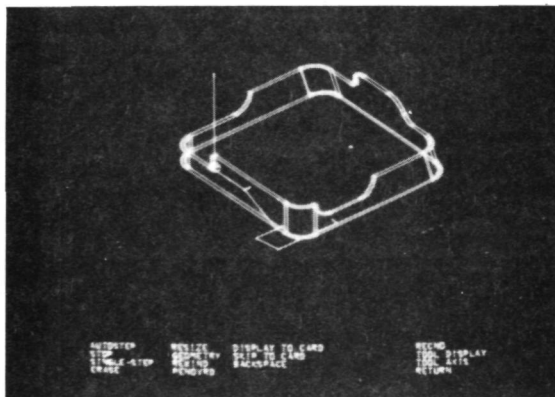




**Fig. 21 DNC Inspection Machine Probing Points on the Surface of the Part**



**Fig. 22 "Blasting" Points on the Surface of the Part Where the Probe is to Make Contact**



**Fig. 23 Plotting the Path the Probe Traverses and the Points of Contact**

During the actual inspection operation, the inspection probe is driven by DNC to each of the preprogrammed inspection points until contact of the part is made. When contact is made, the probe stops, and the actual surface coordinates, the programmed surface coordinates, the allowed tolerances, and out of tolerance conditions are automatically printed out on a high speed line printer. This process eliminates many hours of inspection set-up time and the human error that can result from more conventional inspection methods.

This inspection method is primarily used for first article inspections, for checking out modified GNC programs, or for checking parts to determine the effects of cutter tool wear.

For inspection of subsequent parts after a first article inspection, a Film Inspection Apply Template (FIAT) is sometimes used. It is made of a clear polyester plastic (Mylar) sheet 7.5 mills thick. When a FIAT is required a console operator in the Loft department will call up a copy of the engineering drawing, display it on the CRT, and input a request to the computer for a scribecoat hardcopy. When the hardcopy is completed, a photo-sensitive sheet of Mylar is overlaid on the hardcopy, exposed, and run through a development processor. Within a matter of minutes, an inspection template made of a dimensionally stable-base material is ready for use. These templates are used by Quality Assurance to check a finished machined part for proper machining of flanges, webs, and caps by overlaying the FIAT on the machined part. Any deviation from the intended configuration is readily detectable.

## SYSTEM DESCRIPTION

MCAIR's software packages utilize a user-oriented programming language that is specifically tailored for the types of problems typically encountered in an aerospace engineering and manufacturing environment. They were developed to allow engineers to utilize the speed and accuracy of a large digital computer to assist in solving analytical and geometry-oriented problems and to present engineering data to other disciplines, both engineering and manufacturing, in the form of computer stored data or as a hardcopied engineering drawing. The output of the computer, which consists of geometric figures and relevant information

relating to them, is visually displayed on a CRT.

The engineer can interact with the computer based on what he sees displayed on the CRT or by inputting specific instructions to the computer by means of the functional and alphanumeric keyboards. In essence, he has replaced his drawing board with the display console. Because of the immediate information feedback aspect of the designer's visual relationship with a computer, he is able to complete his tasks in a shorter period of time.

Each of our 15 CADD consoles is remotely connected to, and used in conjunction with, an IBM 360 Model 195 computer. Each console consists of a display CRT, a typewriter-like alphanumeric keyboard for typing in dimensional data or special instructions to the computer, a 32-button functional keyboard for ordering the creation and manipulation of geometric data (points, lines, arcs, conics, cubics, planes, surfaces, etc.) on the CRT, an electronic light pen for detecting specific displayed data to be manipulated, a closure hood with a Polaroid camera for quick-reference pictures, and a "hot-line" telephone to the CPU (Central Processing Unit) operator. Figure 24 shows a typical CADD console arrangement.



Fig. 24 CADD Console Arrangement

The console operator inputs information and orders to the computer, using the various input devices at the console. The output of the computer, which consists primarily of geometric entities, is displayed on the screen of the CRT. Some simple

examples of functions which may be performed using the CADD system should illustrate the types of things that can be done.

For example, one of the ways in which an operator can generate a point is to hold the light pen in close proximity to the screen at the desired location and depress the function key labeled GENERATE POINTS. Figure 25 shows the function keyboard arrangement. A point will be generated and displayed in the vicinity of the location of the light pen tip. The position of the point, while mathematically precise, is only an approximation of the place or point where the light pen is held, however. If the designer wishes to generate a point at a mathematically precise location, he must depress the function key labeled POINT. A message (called a menu) will be displayed on the screen requesting the operator to input the desired 3-D coordinates of the point. With this accomplished using the alphanumeric keyboard, the computer generates and displays the point.

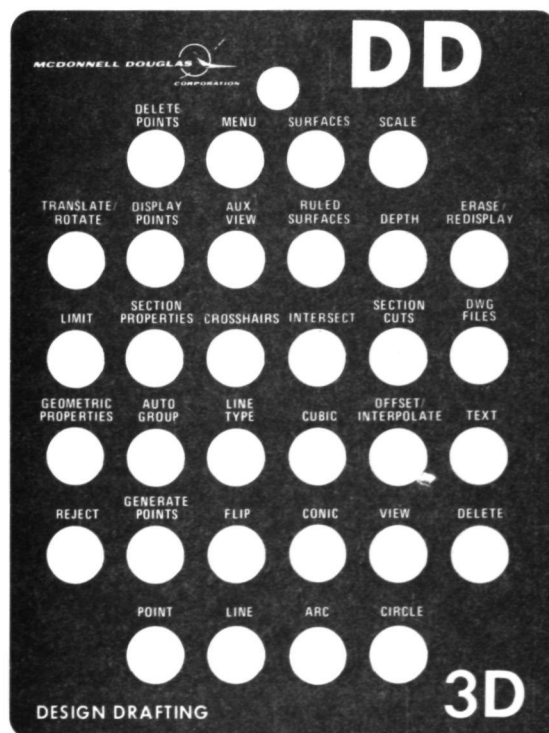


Fig. 25 CADD Function Keyboard Overlay

Although the light pen was used to indicate the approximate position of a point to be generated in the first example, its primary use is in detecting screen images of entities with which the operator wishes to work. Detection is accomplished by positioning the light pen over the image on the screen, then depressing its spring loaded tip against the screen. The light from the image registers on a photocell located within the pen and when detected, the image disappears, indicating that it has been detected. As soon as the operation on the detected element is completed, all detected entities are redisplayed, along with any new ones that have been created.

One way of generating a finite length line is for the operator to detect with the light pen two end points, having any x, y, z coordinates and depressing the function key labeled LINE. (Depression of a specific function key tells the computer what kind of entity it is to generate, i.e., line, circle, arc, etc.). The two detected points give the computer all of the information needed for it to generate the line. The result is a line between the two detected points. Should insufficient data be supplied to the computer for an entity generation, such as detecting one point with the light pen and depressing the function key labeled CIRCLE, a message will be displayed on the screen asking for a radius to be keyed in. Thus, the logic programmed into the system has been well planned in a natural engineering mode.

These examples illustrate how entities may be generated by using the function keyboard and the light pen (GENERATE POINTS and LINE) or the function keyboard and alphanumeric keyboard (POINT), but the more general case of generating geometric entities would utilize all the input devices in combination. For instance, to offset a line parallel to another line, the operator would detect the original line with the light pen and depress the LINE function key. The CRT would then display a menu requesting the distances from, and the side of, the detected line on which the parallel line is to be generated. When that information is supplied via the alphanumeric keyboard and the light pen, a parallel line is generated and displayed.

In addition to points and lines, our graphic system is capable of generating many other kinds of geometric elements. Separate function keys exist for

POINT, GENERATE POINTS, LINE, ARC, CIRCLE, CONIC, ELLIPSE, SURFACES, and CUBIC entities, as shown in Fig. 25. Furthermore, since there are a number of different ways each element may be generated, virtually any figure can be defined by various combinations of the basic geometric elements.

The use of the computer in the design process has resulted in additional benefits and capabilities that enhance the basic attractiveness of the system. Admittedly, one of the chief attributes of computer graphics is its capability to geometrically define an object far more rapidly and accurately on the CRT, with the aid of the computer, than an engineer could on a drawing board. Once a computer is utilized and integrated into a graphic system, it lends itself well to a variety of tasks associated with generating, displaying, and manipulating geometric data. All of the geometric data displayed on the CRT are physical representations of what is modeled in the high speed core memory of the computer.

Once the computer has been programmed to deal with data in terms of a model, it is then possible for the computer to manipulate that model. Data manipulation functions such as SCALE (expand or contract the displayed drawing), FLIP (reflect a display about a line, i.e., mirror image), and TRANSLATE/ROTATE (translate and/or rotate a picture from one position to another) all make it easier for the console operator to work with the display. Beyond these, however, functions such as GEOMETRIC PROPERTIES (calculated geometric properties of selected figures), SECTION CUTS (display the cross-section of the displayed model when it is intersected by a plane), and VIEW (automatically view the displayed model from any desired vantage point) extend the utility of the system to a point where the computer is capable of accomplishing tasks easily which, if they were even possible on the drawing board, would normally be time consuming and often resulting in some degree of inaccuracy. This is one of the initial payoffs of computer graphics.

#### **ADVANCED DESIGN: CONFIGURATION SYNTHESIS**

The CADD system is effective as a design tool in Advanced Design. The aircraft configurationist is able to rapidly and accurately construct the

vehicle, using time saving CADD capabilities such as the automated wing planform routine, the kinematics routine for defining the landing gear and controls spatial mechanism geometry, the vision plot routine for ascertaining the limits of the pilot's vision from the cockpit, the vision of the missile's seeker head when mounted as an external store, and the retrieval of engine and weapons geometry, previously created and stored in the computer files. This eliminates the need for recreating these common geometry data each time they are needed for a different vehicle configuration. The designer is also able to define the surfaced outline of the vehicle, the fuel tanks and other internal volumes. This provides him the capability to take section cuts of the vehicle to determine area distribution curves, fuel volumes, and other properties of the configuration that affect the vehicles's performance and mission adequacy.

The flow chart shown in Fig. 26 illustrates the sequence of events that occurs in advanced design activities. A more detailed description of these activities are discussed in the following paragraphs.

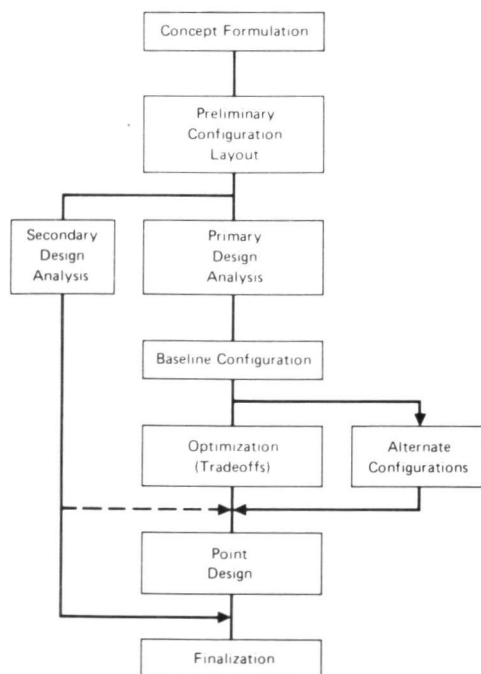


Fig. 26 Advanced Design Flow Chart

## Concept Formulation

Concept formulation is the summation of all the vehicle study requirements and is comprised of such parameters as takeoff gross weight, thrust-to-weight ratio, wing loading, fineness ratio, fuel fraction, payload tradeoffs, general mission requirements, and specific ground rules. The ground rules are normally related to mission and customer specifications and generally pertain to equipment accessibility, materials to be used, design-to-cost requirements, takeoff and landing performance requirements, maintainability, reliability, etc. The other requirements noted are aerodynamic related parameters derived from statistical and theoretical data based on the projected technology of the study's time frame.

## Preliminary Configuration Layout

The aircraft configurationist must graphically establish a moldline definition of the vehicle and locate the various system components and equipment. He must establish a large matrix of vehicles defined by many combinations of the vehicle component variations: high wing versus low wing, side versus bottom inlets, podded engines versus fuselage mounted engines, etc. This is normally time consuming and requires extensive analysis before an optimum configuration can be established. Figure 27 shows a few of the vehicle permutations configured and evaluated (without the use of CADD) in the process of establishing a final point design of the F-15 air superiority fighter.

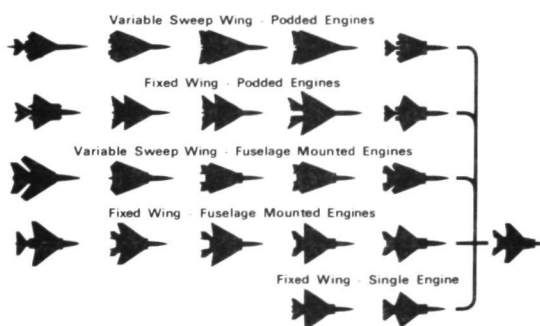
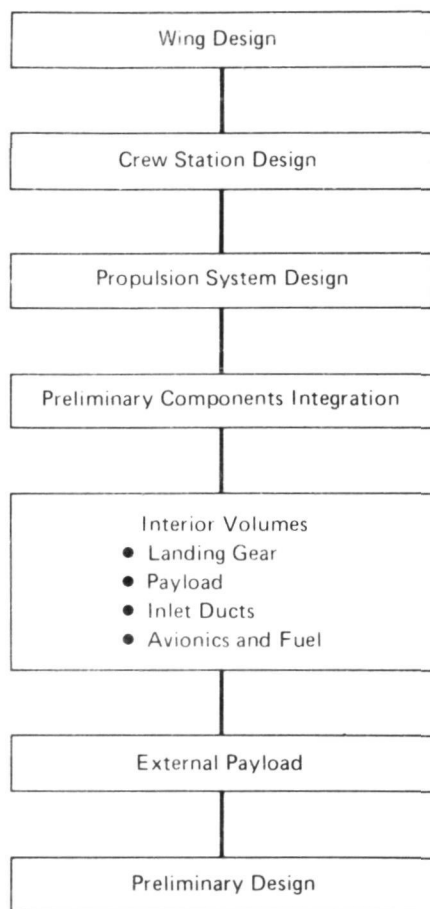


Fig. 27 Configuration Evolution Studies and Tests

A typical sequence of design events, illustrated in Fig. 28, is discussed in the following paragraphs to show in more detail how CADD is currently being used in this process.

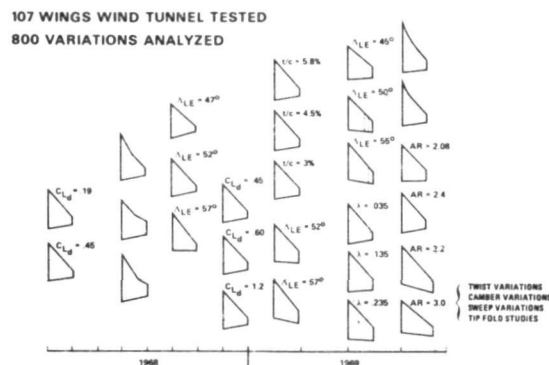


**Fig. 28 Preliminary Configuration Layout Flow Chart**

## Wing Design

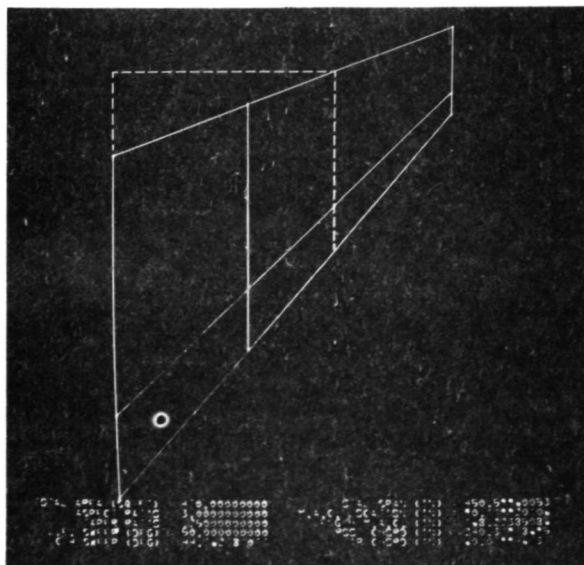
The optimization of a vehicle configuration requires that many different wings be considered and evaluated for performance adequacy. The study analysis, and test of various combinations of geometric considerations such as wing area, aspect ratio, leading edge sweep angle, taper ratio, and variations of wing twist, camber, and sweep are

necessary to assure the best geometric and aerodynamic wing design is selected. For example, Fig. 29 shows but a few of the wing planforms considered for the F-15 Air Superiority Fighter.



**Fig. 29 Wing Development**

Today, with the use of CADD, this design and selection process provides a compression of the study's time frame at less cost. To establish a wing planform, the designer needs only to light pen detect **TAPERED WING** from the menu displayed on the CRT, and type in values for wing area, aspect ratio, taper ratio, sweep angle, and apex origin. The wing planform, mean aerodynamic chord, and quarter chord are immediately created and displayed as shown in Fig. 30.



**Fig. 30 Wing Layout**



Different types of airfoils, previously created and stored in the computer, are available to the designer for merging with the wing planform to give it a three-dimensional definition in a wireframe form. After selecting the desired airfoil, he scales it to the proper chord lengths for the root and tip chords and translates them to their respective positions on the wing (Fig. 31). By detecting the two airfoils with the light pen and depressing the RULED SURFACE function key, the wing becomes fully surface-defined with a parametric ruled surface (PRS).

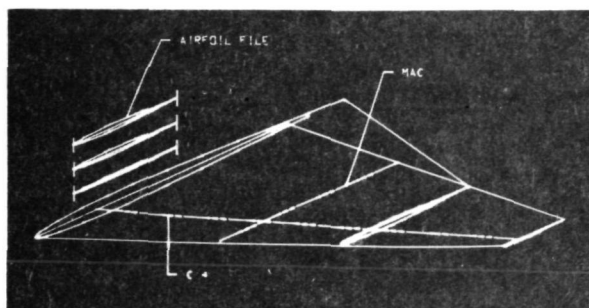


Fig. 31 Wing with Airfoils

Should he desire to define the wing with parametric cubic (PC) patches, he converts the airfoils to PC entities, light pen detects them, and depresses the SURFACE function key. Either of these routines provides the designer the capability to determine the volume of the wing, accomplished by light pen detecting the surface of the wing and depressing the GEOMETRIC PROPERTY function key. Internal volumes, such as fuel tanks, can be determined in the same manner, after defining the tanks with surfaces.

This automated and interactive routine allows the designer to consider many different wings in this phase of the design activities and assures him that he has adequately defined a wing that meets all design and performance requirements.

#### Crew Station Design

The cockpit layout is generally based on specifications, number of crew, method of crew egress, etc. The specification requirements include pilot vision limits, percentile man, instruments required, and cockpit clearance. The methods of egress could be with ejection seat or escape module,

with the selection normally resulting from trade-offs between survivability requirements and minimum weight of the system.

By positioning the pilot's eye at the desired location, the designer is able to construct the canopy and establish the fore, aft, and over-the-side vision constraints as shown in Fig. 32. With this established, he can then construct the sill width, console width, and sidewall thickness. The forward fuselage can then be completed by constructing the fuselage half breadth line, the upper and lower moldline, several fuselage station cuts, then transforming the wireframe geometry into a fully surfaced definition in a manner similar to that previously discussed for the wing design.

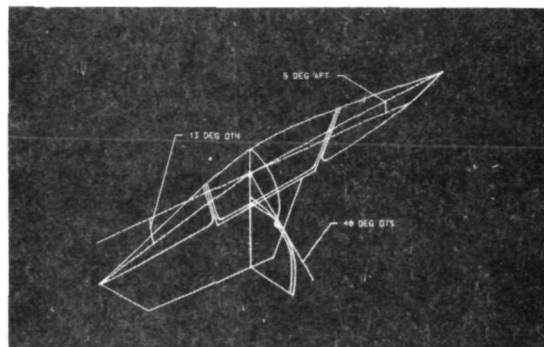
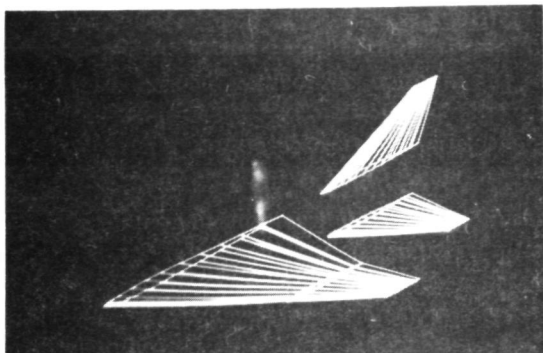


Fig. 32 Cockpit Layout

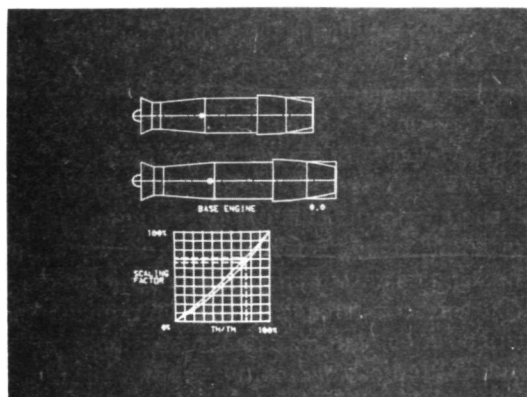
#### Systems and Structural Component Integration

The remainder of the fuselage structure and other structural and system components comprising the airplane are then constructed and integrated into a complete definition of the airframe. These components include the tail surface, inlet and propulsion system, landing gear system, fuel tanks, avionics bays, and weapons systems. The tail surfaces are constructed in the same manner used for constructing the wing. Figure 33 shows the wing and tail surfaces. For the propulsion system design, the designer must first size the engine based on aerodynamic inputs (T/W versus TOGW) using propulsion scaling curves as shown in Fig. 34. The base engine configuration, defined and stored in the computer, is retrieved from the computer files, displayed on the CRT, and translated to a position established by Weights department personnel in their preliminary balance of the vehicle. The inlet

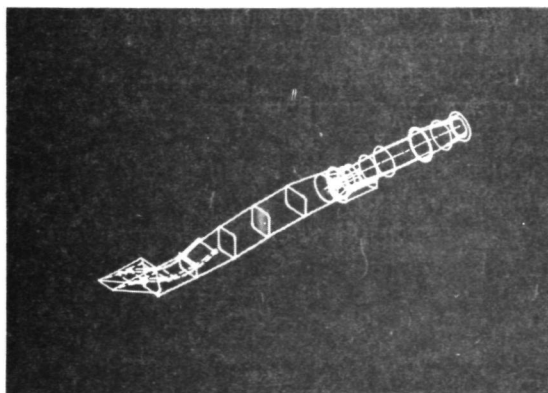
system is then constructed and merged with the engine as shown in Fig. 35.



**Fig. 33 Ruled Surface of Wing and Tails**

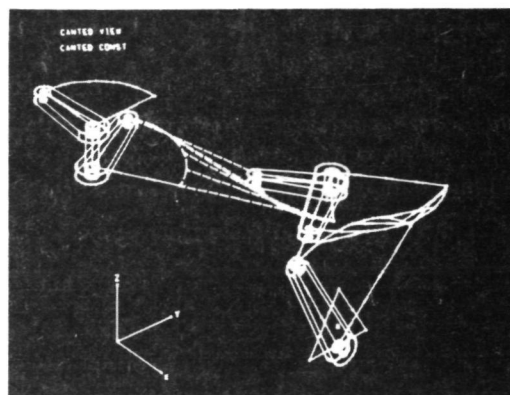


**Fig. 34 Engine and Sizing Curves**



**Fig. 35 Propulsion System**

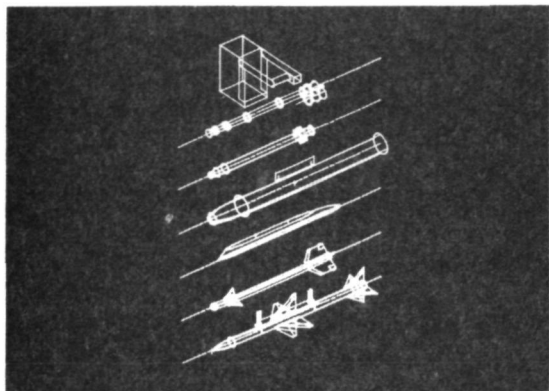
The kinematics routine in the CADD system is ideally suited for solving landing gear mechanism problems. The location of a skewed axis landing gear trunnion can be determined readily by establishing the extended and retracted positions of the landing gear wheel. In addition, the path of the gear during retraction is displayed in as many increments as desired, allowing the designer to determine landing gear door opening requirements and minimum clearances with aircraft structure and external stores in the landing gear retraction/extension cycle. It is also beneficial for solving spatial four bar linkage problems (Fig. 36) that are time consuming and fraught with tolerance errors when solved manually. This routine provides a graphic display of the positions of the driver, coupler, and driven bellcrank of a spatial mechanism with its intermediate positions displayed, thus providing the designer mechanical advantages and load/stroke data for the mechanism throughout its motion, merely by interrogating the computer.



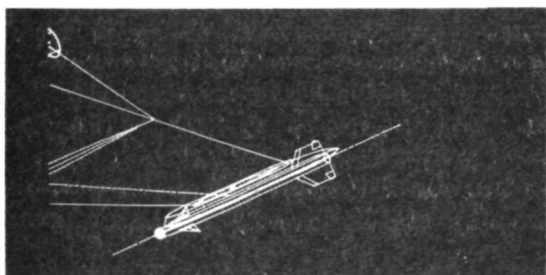
**Fig. 36 Mechanisms**

Several types of external and internal weapons are always considered when defining combat mission scenarios for military aircraft. In the past, it was necessary to trace or redraw the weapons at various locations on the wing and fuselage for each of the various tradeoff configurations, which was a time consuming and non-creative task. Currently, the designer can retrieve from the computer the weapons file (Fig. 37) that contains various military inventory weapon geometric descriptions, and merge the selected weapons with the vehicle

description displayed on the CRT at any location he desires. Figure 38 shows a missile that has been placed at the airplane's wing tip. This capability saves countless hours of prosaic drafting that would be required if computers were not used.



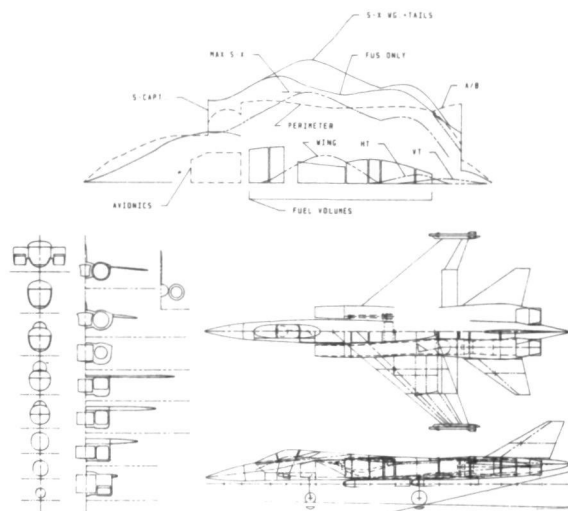
**Fig. 37 Weapons File**



**Fig. 38 Missile Placement**

When the "preliminary design" (Fig. 39) configuration is completed, it is ready for aerodynamic analysis to determine if its mission and performance requirements can be met. If analysis shows that these requirements have not been met, ICADE may be used to size the vehicle and determine the sensitivity of the aircraft size to incremental changes in the various vehicle components.

With the rapid output of these data from the computer to the advanced design team, a more thorough evaluation of the study vehicle can be made, attributed largely to the increased number of design iterations possible in a given time frame, and due to increased visibility and recognition of convergence to the desired performance results.



**Fig. 39 Preliminary Configuration and Area Distribution**

### Wave Drag Analysis

The IBM 2250 graphics console has been adapted to provide the geometric input data deck for the NASA Wave Drag Program where these input data are coordinates of surface points, surface chordal lengths, surface thickness ratios, etc., for a particular aircraft configuration that has been created by the designer and filed in the computer. The CADD module is used to make section cuts through the surfaced configuration of the vehicle (Fig. 40) and to create boundary points on those section cuts (Fig. 41). The points are then connected with straight lines, labeled, and filed in the CADD drawing files (Fig. 42). The aerodynamics analyst then accesses the section-cut/point data from the CADD files and determines and stores  $x$ ,  $y$ ,  $z$  coordinates of the points on an on-line disk pack in a card image form (Fig. 43). In order to complete the procedure for obtaining the geometric input data deck for the NASA Wave Drag Program, it is necessary to utilize an auxiliary computer program on the IBM 360 computer to access the on-line disk pack geometric data, reorient them, and calculate the geometric data in the format required by the NASA Wave Drag Program. It should be noted that this data deck can also be used for the Subsonic Potential Flow Program, the Subsonic-Supersonic Linear Theory Program, and the



Supersonic-Hypersonic Arbitrary Body Program, all of which are active, analytical computer programs used at MCAIR. Figure 44 shows a schematic of the data flow for the NASA Wave Drag Program.

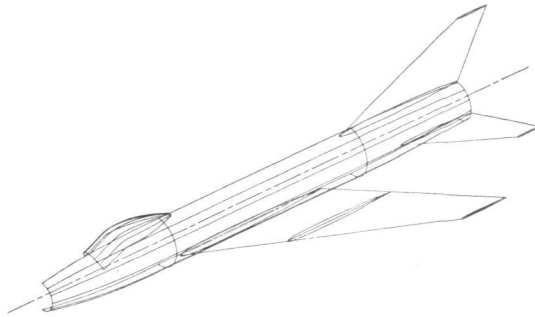


Fig. 40 Surfaced Vehicle Configuration

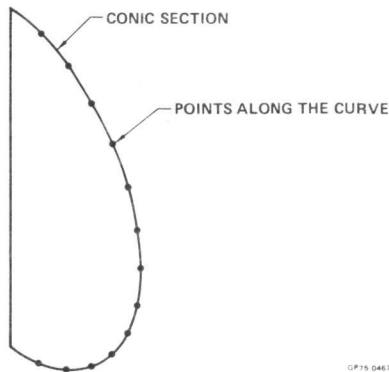


Fig. 41 Section Cut with Boundary Points

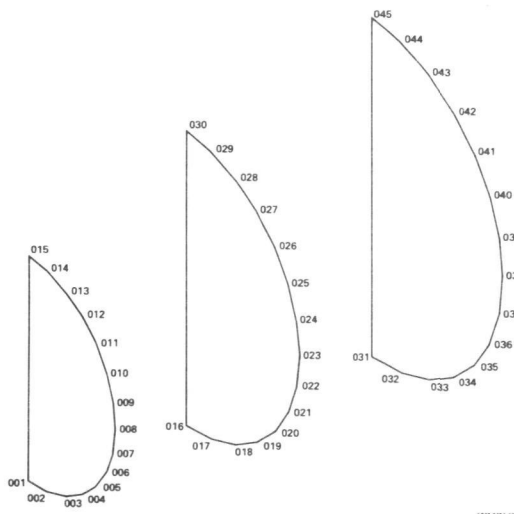


Fig. 42 Points Connected with Straight Lines and Labeled

POINT	X	Y	Z
10001	.00000	.00000	12.0000
10002	.00000	19.36246	6.40355
10003	1.39400	19.36246	6.52022
10004	2.75050	19.36246	6.86749
10005	4.02853	19.36246	7.43716
10006	5.19116	19.36246	8.21519
10007	6.20235	19.36246	9.18134
10008	7.02898	19.36246	10.3088
10009	7.64245	19.36246	11.5640
10010	8.02052	19.36246	12.9063
10011	8.15028	19.36246	14.3119
10012	8.03072	19.36246	15.7667
10013	7.67538	19.36246	17.1793
10014	7.09197	19.36246	18.5128
10015	6.29330	19.36246	19.7288
10016	5.29835	19.36246	20.7895
10017	4.13314	19.36246	21.6592
10018	2.83122	19.36246	22.3058
10019	1.43337	19.36246	22.7038
10020	.00000	19.36246	22.8366
10021	.00000	38.72492	3.57100
10022	2.29010	38.72492	3.76551
10023	4.51441	38.72492	4.34410
10024	6.60814	38.72492	5.29188
10025	8.50875	38.72492	6.58376
10026	10.15756	38.72492	8.18422
10027	11.50173	38.72492	10.0473
10028	12.49655	38.72492	12.1174
10029	13.10793	38.72492	14.3293
10030	13.31527	38.72492	16.6238
10031	13.11953	38.72492	18.9777
10032	12.53657	38.72492	21.2667
10033	11.57942	38.72492	23.4263
10034	10.27068	38.72492	25.3928
10035	8.64338	38.72492	27.1049
10036	6.74177	38.72492	28.5058
10037	4.62148	38.72492	29.5459
10038	2.34855	38.72492	30.1864
10039	.00000	38.72492	30.4024
10040	.00000	67.00000	1.44415
10041	2.98750	67.00000	1.68090
10042	5.90019	67.00000	2.38556
10043	8.66482	67.00000	3.54149
10044	11.21101	67.00000	5.12110
10045	13.47289	67.00000	7.08602

Fig. 43 Card Image Form of Point Coordinates

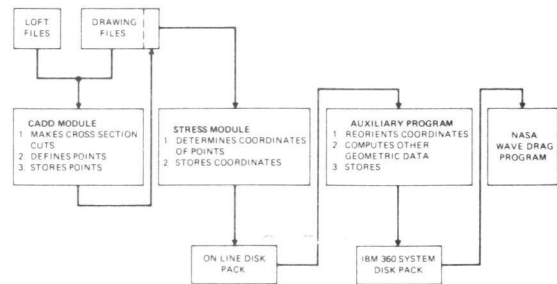
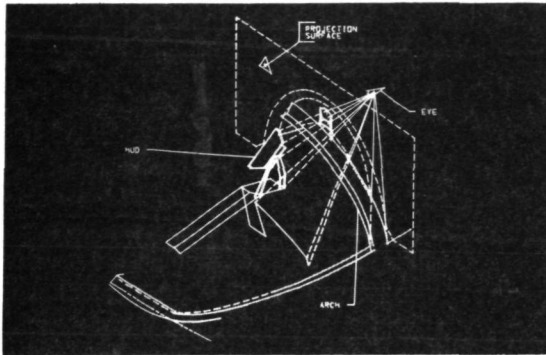


Fig. 44 Schematic of the Procedure for Obtaining the Data Deck for the NASA Wave Drag Program Using the IBM 2250 Graphics System

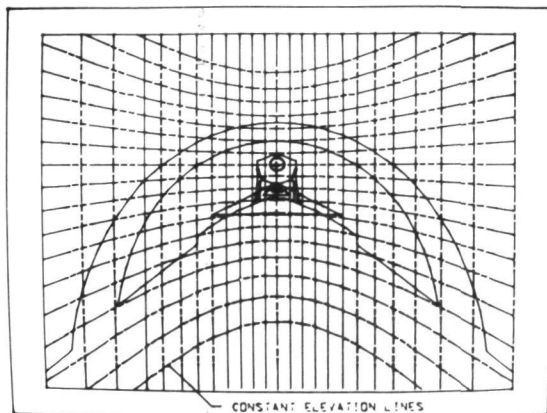
#### Field of Vision Plots

Investigation of the pilot's field of vision is a customer requirement. In the past, the results were illustrated on two-dimensional diagrams culminating from geometry data generated by conventional drawing board techniques. With the construct capabilities and the three-dimensional

obtain a comprehensive cross section for point design selection. The baseline designs are investigations of tradeoffs of such things as variations of wing sweep and aspect ratio (Fig. 48), and fixed versus variable sweep wing geometry (Fig. 49).

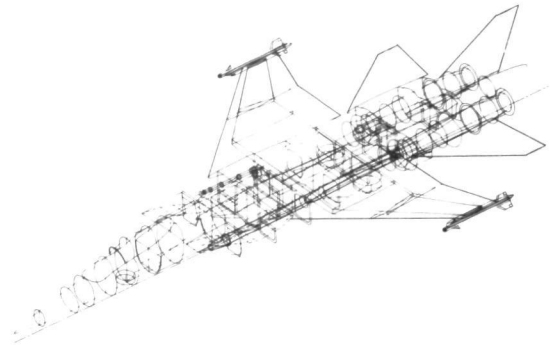


### Fig. 45 Vision Plot Construction

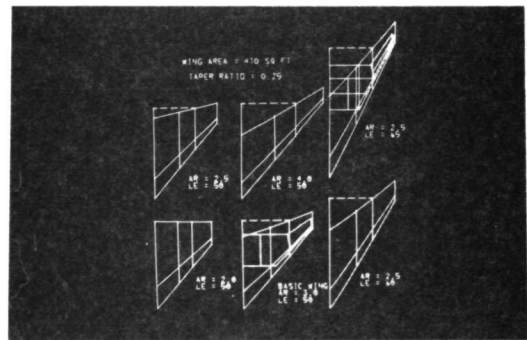


**Fig. 46 Completed Vision Plot**

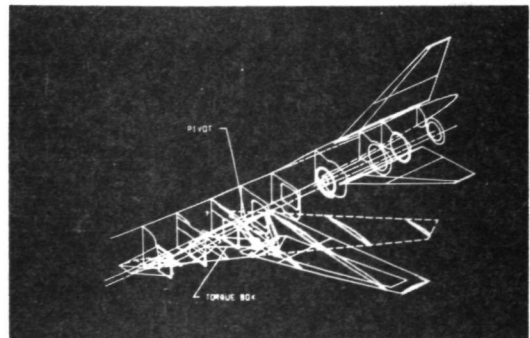
The configuration at the end of the optimization cycle becomes a point-design configuration (Fig. 47), and copies of the configuration may be retrieved by the designers from the computer files for auditing or for making more detailed design studies, such as internal structural arrangements, landing gear mechanism geometry synthesis, field of vision variations, etc. The point designs are representative of established mission areas in the study matrices. There are as many baseline designs generated as are optimized as are practical to



**Fig. 47 Trimetric of Point Design Configuration**



**Fig. 48 Wing Geometry Tradeoff Study**

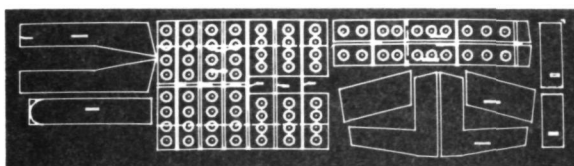


**Fig. 49 Variable Geometry Configuration**

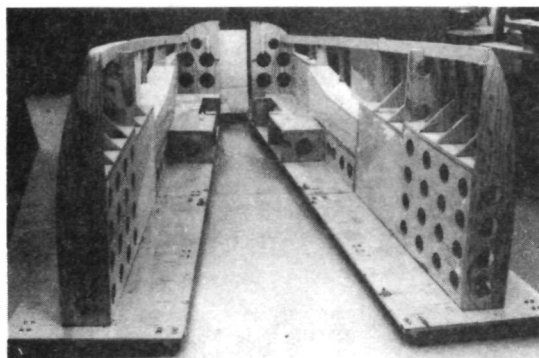
Some points in a study matrix may be completely different design approaches to preclude the omission of any configuration possibility. In the end, the program manager must select the vehicle configuration that best meets the mission requirements stipulated by the customer and establish it as the "point design".

### Design Aids

Design aid models can be made from layouts constructed on the CRT using the filed configuration, with the resultant drawings being transferred from the CADD system to paper by an automated plotter (Fig. 50). The paper patterns can then be glued to a material, such as plywood, cut out, and assembled (Fig. 51).



**Fig. 50 Pattern Layout for Cockpit Design Aid**

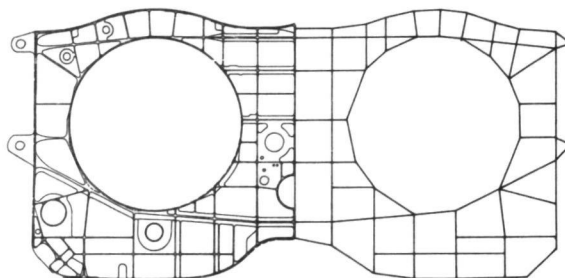


**Fig. 51 Plywood Design Aid of Cockpit Area**

## PRODUCT DESIGN APPLICATIONS

When the designer has the desired lofted surface contour displayed on the CRT, he is ready to start the detailed design of his loft surface related part. For example, in the case of a fuselage-station cut at a bulkhead or frame location, the designer accesses the loft files, types in the pertinent loft surface identification information, and momentarily, the desired moldline contour of the fuselage-station cut is displayed on the CRT. He may then create a

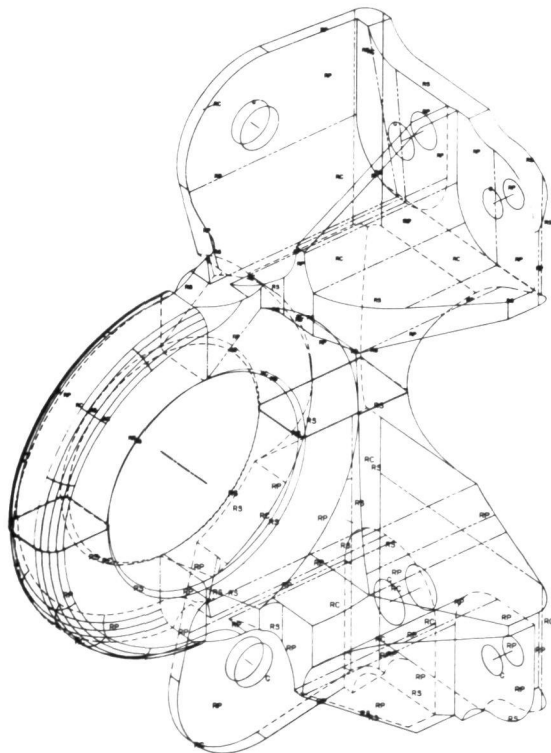
line, offset from the moldline contour, distance equivalent to the thickness of the fuselage sheet metal skin, resulting in having displayed a complete outside definition of the part to be designed. This saves hours of drafting time and eliminates the need for conventional splining techniques which often result in deviations from the desired part outline. After the periphery of the part has been displayed, a finite element model may be created by the strength engineer utilizing the CGSA module. Figure 52 shows a merged hardcopy of a typical fuselage bulkhead including a finite element model and its completed design configuration. This activity is accomplished after having coordinated with the designer during the layout of the part (either on the drawing board or at the console), thus establishing various design constraints that might dictate the locations of the major load paths.



**Fig. 52 Finite Element Model (Right) Merged with Final Design Configuration (Left) of a Typical Fuselage Bulkhead**

Constraints such as holes and openings for engine ducts, lines for fuel, ECS, and hydraulics, and mounting provisions for various pieces of equipment that are to be installed are examples. After the load paths are established, the finite element model established, and the member loads and stresses displayed, the designer is ready to size the thicknesses of the flanges, caps, and webs of the part. With these items properly sized, he creates a three-dimensional wireframe model of the part, surfaces it with parametric ruled surfaces or parametric cubic patches, automatically creating unique labels for each surface, and takes appropriate section cuts of the part. Figure 53 shows a surfaced and labeled part. The labels are identifiers used for accessing specific portions of a model stored in the computer. This provides manufacturing planners a quick method of accessing data

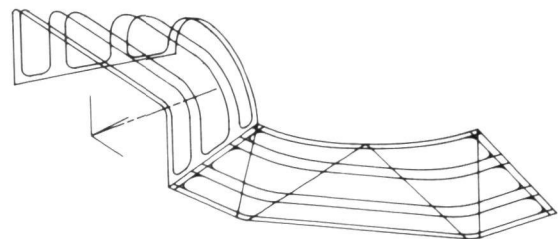
when they are creating the APT program for machining the part. Certain portions of these section cuts are then dimensioned, using the automated dimensioning routine, to aid in the final stress analysis of the part. This dimensioning routine, which creates and displays dimensions to areas of the part that have been light pen picked, eliminates the errors that might occur if the operator manually keyed-in the dimensions via the alphanumeric keyboard. Even though certain section cuts have dimensional data displayed, a major portion of the geometry of the part will not have displayed dimensions, since the entire part is completely defined mathematically in the computer, providing all the numerical data necessary to machine the part using numerically controlled milling machines.



**Fig. 53 Surfaced and Labeled Part**

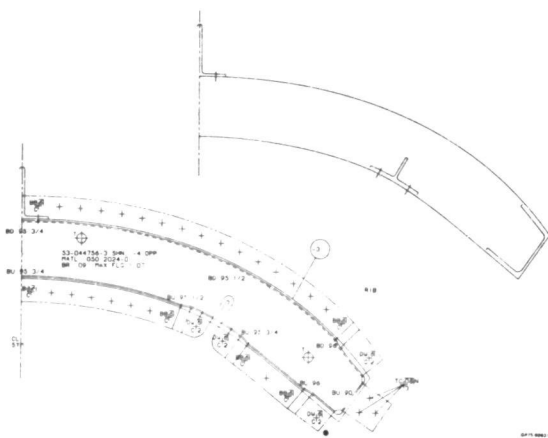
CADD has also been an invaluable tool in the design of laminated composite structure such as the F-15 speed brake. It is made up of a series of 0.1372mm thick graphite composite plies that are pre-impregnated with epoxy. There are

seventy-six plies at one section of the speed brake, each of which is moldline related. In the past, it was necessary for the designer to describe each ply in the installed configuration, but he was unable to define the flat pattern dimensions required by manufacturing to cut out the plies from raw stock. Personnel from the Master Layout section of the Loft department would have to manually lay out each ply, which was costly and time consuming. With the Triangulated Flat Pattern routine in the CADD system, our designers now define the installed, curved configuration of each ply, then instruct the computer to unroll them into a flat pattern. In an instant the flat pattern of the ply is displayed on the CRT screen. Each of the flat patterned plies is then hardcopied at full scale and input to manufacturing as a template for cutting out the material. Thus, manual layout errors are eliminated and the design/fabricate cycle time is reduced. Figure 54 shows a typical example of a flat pattern that has been rolled out from its curved installation position.



**Fig. 54 Triangulated Flat Pattern**

Conventional sheet metal parts are easily defined with CADD. By light pen detection, a CADD operator can input the contour and flange limits of the part. In turn the computer asks tutorial questions that may be answered by selecting menu items displayed on the CRT screen or by the alphanumeric keyboard. The variables requiring answers are flange width, material thickness, flange bend radius, and rivet size and spacing. From these inputs, the program develops a completed flat pattern of the part showing joggles, developed flanges, rivet locations, bend angles, form block lines, and bend tangent lines. This routine compresses the design time required for sheet metal parts. Figure 55 shows a typical flat pattern sheet metal part designed with this routine.



Many other types of design tasks are accomplished daily using the CADD system, including complex spatial kinematics synthesis problems, the routing of hydraulic and fuel lines, general equipment installations, determination of clearance problems of moving elements with respect to structure, and generally, any type of layout work conceivable. All of these tasks are accomplished faster and with more fidelity than was ever possible prior to the advent of interactive graphics.

After parts have been designed utilizing the CADD and CGSA modules and released by engineering, manufacturing must concern themselves with the task of using the stored geometric data to provide computerized instructions for driving the N/C milling machines to cut the parts. These tasks are accomplished with the Graphical Numerical Control system, using the same type console used by design engineering to define the parts. Using GNC, Automatically Programmed Tool (APT) source statements are created for operating five-axis machines as well as a Direct Numerical Control (DNC) precision measuring and inspection machine.

which uses a high speed disk file for program storage and is capable of driving 10 DNC machine tools simultaneously. The machine tools are different configurations and can cut different parts at the same time.

## SUMMARY

# APPLICATIONS OF COMPUTER GRAPHICS TO AIRCRAFT SYNTHESIS

By Ralph L. Carmichael\* and Richard Putnam\*\*

NASA Ames Research Center

## SUMMARY

The results of the aircraft design process are best given in graphical, pictorial, and tabular form. The computer-based aircraft synthesis programs currently being developed will generate this graphical output. The Advanced Vehicle Concepts Branch, NASA-Ames, has been implementing an aircraft synthesis program with computer graphics since 1971. This paper describes the history of the program development and compares a system based on time-sharing to two different concepts based on distributed computing.

## INTRODUCTION

Increasing automation in the airplane design process is an important subject of research within the government and aviation industry (refs. 1-11). The implementation of such plans will call for eventual widespread use of computer-generated graphical output. The purpose of this paper is to relate the experience of the Advanced Vehicle Concepts Branch, NASA Ames Research Center, in implementing computer-generated graphics in conjunction with the program for aircraft synthesis (ACSYNT).

## AIRCRAFT SYNTHESIS

There is a substantial effort in the aeronautical research community to develop a computer-based procedure for the synthesis of aircraft configurations (refs. 1-8). The first task in this research effort is to identify and model all first-order effects on aircraft configuration definition. These mathematical models are coded into procedures in the various technical disciplines, such as aerodynamics or propulsion, and coupled through a common data base. By appropriate control logic, an aircraft configuration is synthesized that satisfies the desired mission objectives. This computerized, preliminary design process can provide important information that can influence the direction and magnitude of the overall airplane project at considerably less cost in time and money than a traditional preliminary design.

At the Ames Research Center, the Advanced Vehicle Concepts Branch uses such a synthesis program (ACSYNT) for conceptual design studies of proposed aircraft. The function of an aircraft design capability within NASA is to define quantitative measures of the performance of

---

\*Research Scientist

\*\*National Research Council Post-doctoral Fellow



aeronautical concepts. This permits judgments to be made on the value of various research projects competing for the limited resources available and provides information for the direction and support of NASA aeronautical research.

## GRAPHICS IN AIRCRAFT SYNTHESIS

Since the results of a traditional design study are presented in the form of charts, graphs, drawings, etc., the computer-based study should yield these same results (figs. 1-12). The process of generating this graphical data can be very demanding of engineering time. In view of the capabilities of modern computer graphics hardware, the most rapid and accurate way to produce such graphical output is to include it in the computer synthesis program.

A fundamental guideline, established early in the development of ACSYNT, is that intermediate and final results should be displayed or plotted in graphical form. A second guideline is that the quality of figures be sufficiently high to allow their use in reports without retouching or redrawing. The selection of graphics hardware has been guided by the requirements that graphics be available quickly and that permanent copies be obtainable.

## ACSYNT PROGRAM STRUCTURE

The program structure is illustrated in figure 13. Since one of the ACSYNT guidelines is to avoid machine dependence, the picture creation commands are not directly contained in the discipline-oriented modules. Instead, each module produces an external file containing all the data necessary to create the desired pictures. Upon convergence of the design algorithm, the various files are merged into one combined file of results which is then cataloged and is available for presentation to the user. This is a numerical file and not a graphics file at this point. The graphics post-processor routines are invoked to create and format the plot file of graphical information. This file is written in a specially coded form that can be translated into pictures by any of several devices.

This structure has remained relatively constant over the past few years although the methods for implementing the actual transmission and display of the pictures to the user have undergone several major modifications to adapt to the changing computing environment. The modularization of the program has been very valuable in enabling the adaptation.

## HISTORY OF SYSTEM DEVELOPMENT

The implementation of graphical display for ACSYNT was based on the following assumptions:

1. The user wishes to see the results as quickly as possible after initiation of the job.
2. The user wishes to maintain a certain amount of interactive control over what displays are shown.

3. The user wishes to obtain permanent copies of the graphics and charts but is willing to wait 1-2 days for these if the CRT-type display is available and recallable.

The original implementation utilized a CRT-display computer (an IMLAC PDS-1) as a terminal to a time-shared host computer (an IBM 360/67) (fig. 14). The ACSYNT program was executed on the 360 and created the data file in the central on-line storage of the 360. The displays were generated by a program that was executed in the 360 utilizing a graphics language that could drive both the IMLAC display and the microfilm plotter (an SC4020) to satisfy the requirement for hardcopy. The coding of the display program allowed the user to inspect each plot and then proceed to the next one.

This system was implemented, but the total workload on the time-shared host computer was sufficiently heavy that the action seen by the user was very slow and unresponsive. When optimization of the code and operational procedures did not alleviate the problem, it was decided to abandon the concept of interactive graphics for ACSYNT until sufficient resources could be guaranteed on a time-sharing system.

In foregoing interactive operation of the program, the problem of presenting graphical output was reexamined. The original implementation allowed plots to be made on the microfilm plotter, but the time delay in obtaining the prints was excessive. An alternate approach was devised that enabled the time-sharing computer to generate a picture file that was sent to a graphics support computer (an IBM 1800). This graphics support computer operates a file-management system, enabling the user to catalog a collection of graphical data for later recall. The computer response was still slow and unresponsive, but the user was not required to be in constant attendance, since there was no longer any interaction. The pictures were generated individually and stored on the disk file of the 1800 with no pause between frames. At the conclusion of the run, the user could return to the terminal and redisplay any or all of the frames on a selective basis. This retrieval system is rapid enough to satisfy the user response requirements. This concept has now been carried one step further by attaching a local disk file to the display device (IMLAC). With this facility, the pictures may be retrieved even if the support computer is temporarily unavailable. A mechanical plotter driven by the IMLAC computer enables pen and ink copies to be made at the user's site. This mode of operation, sometimes referred to as "fast-batch," has been the principal technique for ACSYNT graphics up to the present time (fig. 15).

### ACSYNT SUPPORT COMPUTER

The system as described in the previous section represents a good adaptation to the computing environment as it existed when the system design was made. However, during the implementation of the system, the facilities available underwent a significant change, and the graphics support has been redesigned to adapt.

The previously described system was based on the assumption that the ACSYNT program would be executed on the time-shared system (the IBM 360/67). Because this computer was so overloaded, time was made available on a Control Data 7600 located at the Lawrence Berkeley Laboratory (LBL) and accessed through a remote job entry (RJE) station at Ames. This meant that two versions of ACSYNT were kept in operation—one on the 360 with graphics and one on the



7600 without graphics. As seen by the user, the performance of the 7600 was so superior in terms of cost and turn-around time that all the jobs were run there. The advantage of having graphical output could not overcome the disadvantages of long running time and slow turn-around.

Early in 1975, the LBL arrangement was replaced by the installation of a similar machine at Ames. Since it was apparent that this machine would be the site for execution of ACSYNT for several years to come, a design study was initiated to find a way to make the graphical output available.

The Cyber 76 machine installed at Ames does not support interactive terminals in that way that the 360 does, but it does support a network of RJE stations at various points about the laboratory. The most direct approach to obtaining graphical output at the remote site is to have a combined graphics/RJE station. Such a station has been installed and is now being brought into operation. The station is based on a minicomputer (PDP-11/40) and utilizes a high-quality, three-dimensional CRT display (Evans & Sutherland Picture System), local disk and tape storage, and an electrostatic printer-plotter (fig. 16). An interface to the previously described IMLAC PDS-1 is also available. The three-dimensional features of the Evans & Sutherland display enable the aircraft to be viewed from any angle, thereby relieving the host machine of the burden of computing a large number of views.

In the operational scenario for this station, the user prepares an input case for ACSYNT on the local minicomputer and enters this into the input job queue on the host machine (7600) (fig. 17). The job is executed by the host; the user observes this fact via the usual remote station inquiry/response protocol. The file of computed results is transmitted to the minicomputer disk. The pictures are composed and displayed as many times as necessary for insight and understanding to be conveyed to the user while the host machine has proceeded to other computational tasks. After detailed study of these results, the user may wish to change his input data and resubmit the case from the minicomputer station. The process is repeated over and over until the desired study result is obtained. The number of complete turn-arounds accomplished each day depends on the job time and priority. By splitting the workload—giving the host only those functions requiring a large-scale computer and giving the local minicomputer the remainder—the total study time can be minimized.

## CONCLUSIONS

The results of the aircraft design process are best given in graphical, pictorial, and tabular form. It is an essential requirement that as many of the plots as possible be computer generated. Early attempts at the NASA Ames Research Center were based on interactive generation of the plots on a time-sharing system. This general approach has proven infeasible, and two alternative approaches based on rapid retrieval of pictures from batch runs have been implemented. These systems, which are based on distributed computing, are more practical than the former approach until adequate interactive facilities can be made available.

## REFERENCES

1. Gregory, Thomas J.: Computerized Preliminary Design at the Early Stages of Vehicle Definition. AGARD Conference Proceedings 147, Florence, Italy, Oct. 1-4, 1973.
2. Boyles, Richard Q.: Aircraft Design Augmented by a Man-Computer Graphic System. *J. Aircraft*, vol. 5, no. 5, Oct. 1968, pp. 486-497.
3. Lee, Vernon A., Ball, H. Glenn, Wadsworth, E. A., Moran, W. J., and McLeod, J. D.: Computerized Aircraft Synthesis. *J. Aircraft*, vol. 4, no. 5, Oct. 1967, pp. 402-408.
4. Gregory, Thomas J., Petersen, Richard H., and Wyss, John A.: Performance Tradeoffs and Research Problems for Hypersonic Transports. *J. Aircraft*, vol. 2, no. 4, Aug. 1965, pp. 266-271.
5. Williams, Louis J.: Transonic Transport Study - Summary. NASA TM X-62,156, May 1972.
6. Baals, Donald D., Robins, A. Warner, and Harris, Roy V., Jr.: Aerodynamic Design Integration of Supersonic Aircraft. *J. Aircraft*, vol. 7, no. 5, Oct. 1970, pp. 385-394.
7. Hague, D. S., and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles - ODIN/MFV. Technical Report AFFDL-TR 72-132, Aerophysics Research Corp., Dec. 1972.
8. Schuberth, E. R., and Celniker, Leo: Synthesizing Aircraft Design. *Space/Aeronautics*, April 1969, pp. 60-66.
9. Garroq, Carlos A., and Hurley, Michael J.: "The IPAD System: A Future Management/Engineering/Design Environment." *Computer*, April 1975, pp. 23-33.
10. English, C. H.: Interactive Computer Aided Technology: Evolution of the Design/Manufacturing Process. MCAIR paper 75-009. Presented at AIAA 7th Aircraft Design, Flight Test and Operations Meeting, Los Angeles, Aug. 1975.
11. Feder, Aaron: Test Results of Computer Graphic Productivity for Aircraft Design and Fabrication. AIAA Paper 75-967. Presented at AIAA 7th Aircraft Design, Flight Test, and Operations Meeting, Los Angeles, Aug. 1975.

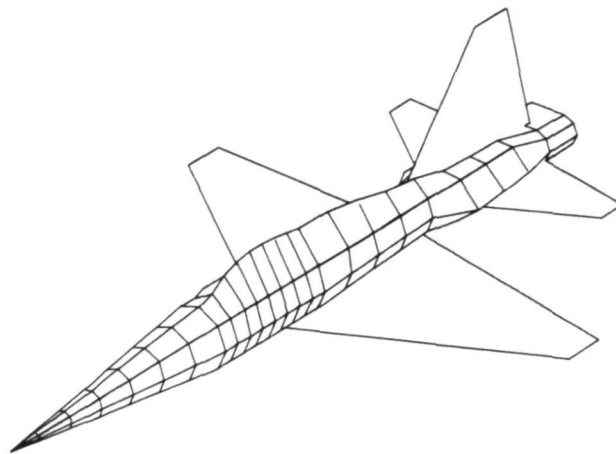


Figure 1.— Perspective view of F-5.

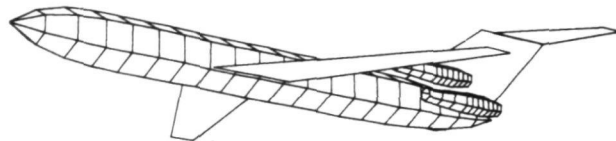


Figure 2.— Perspective view of B-727.

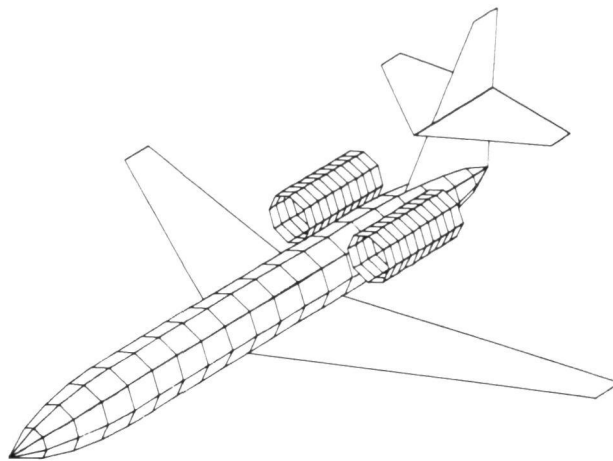


Figure 3.— Perspective view of AWACS escort.

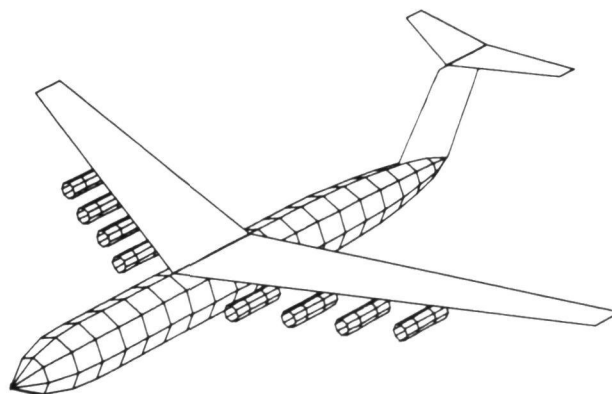


Figure 4.— Perspective view of ATLAS airplane.

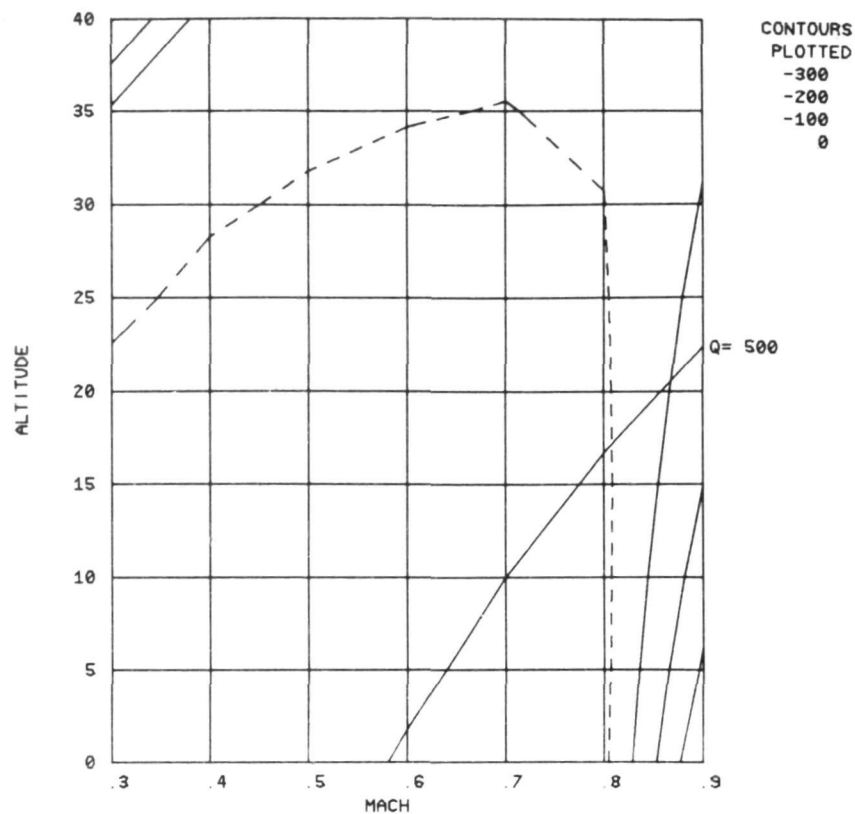


Figure 5.— Contours of constant specific excess power.

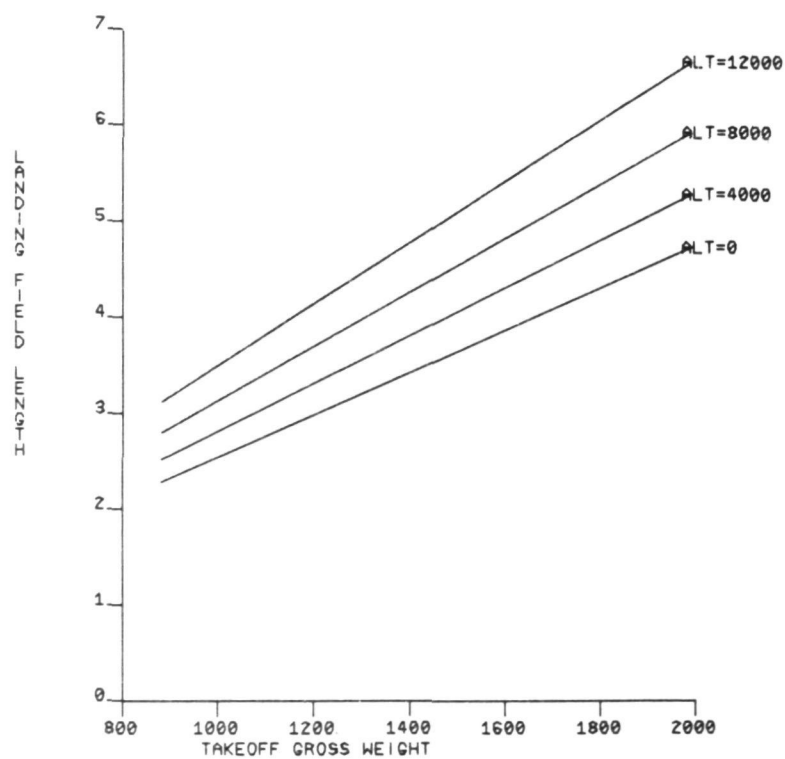


Figure 6.— Landing field length chart.

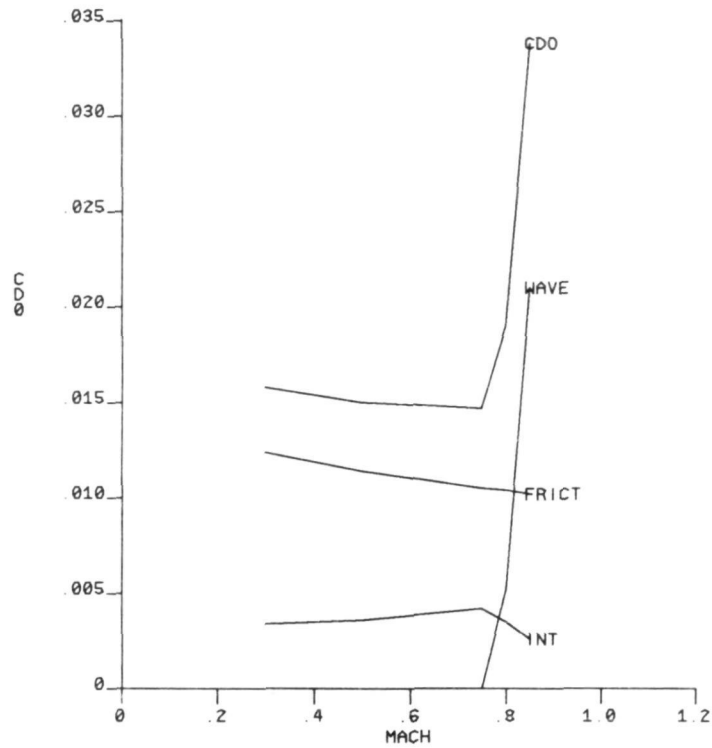


Figure 7.— Zero lift drag buildup.

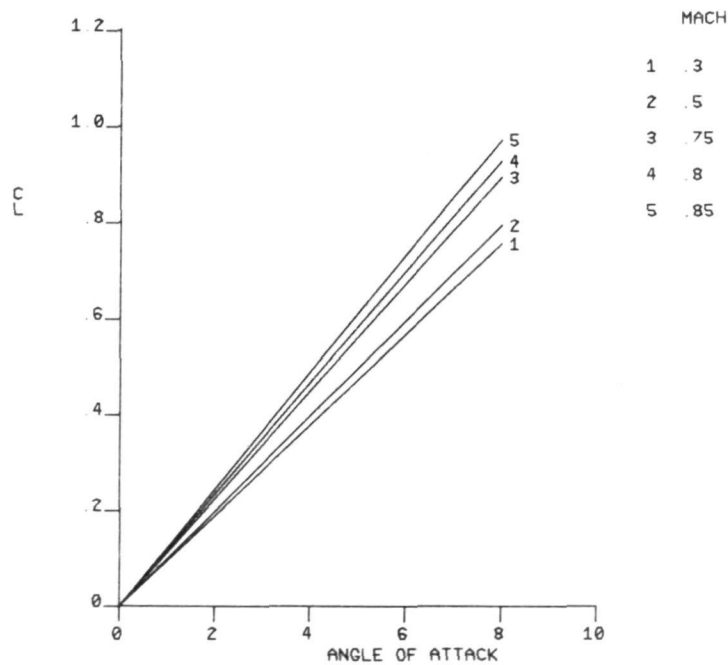


Figure 8.— Lift curves for various Mach numbers.

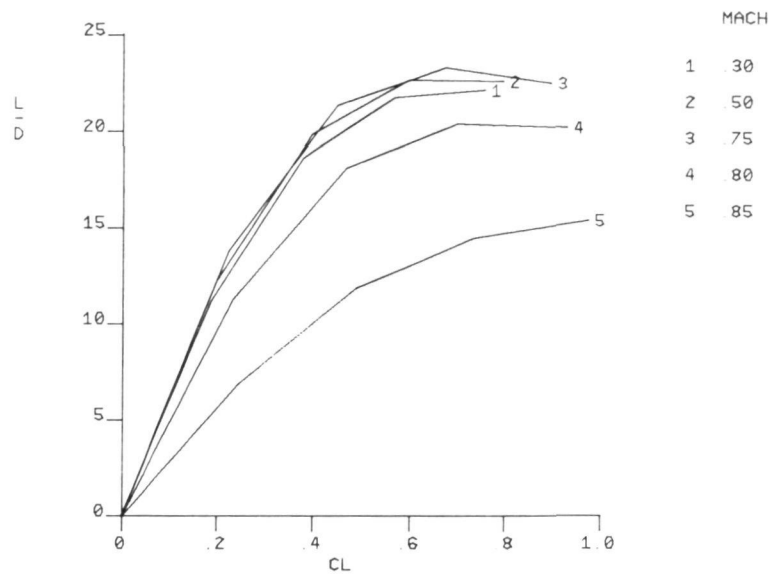


Figure 9.— Lift/drag curves for various Mach numbers.

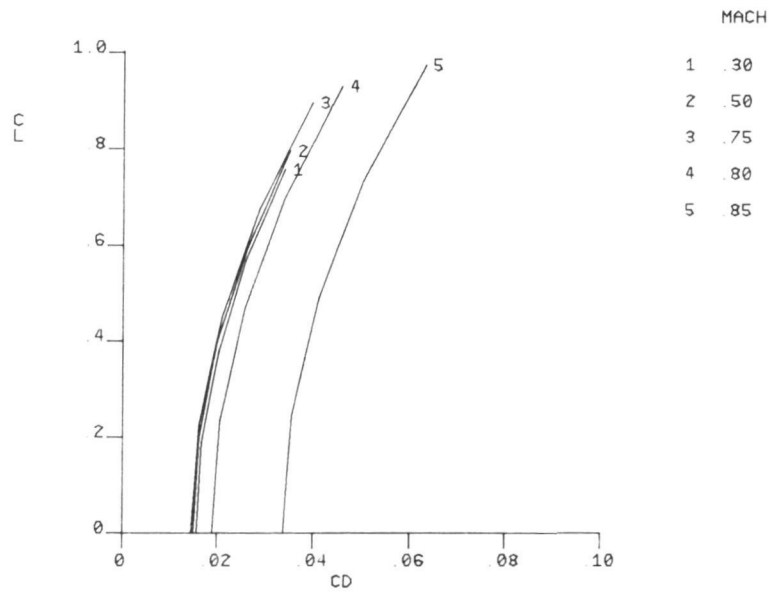


Figure 10.— Drag polars for various Mach numbers.

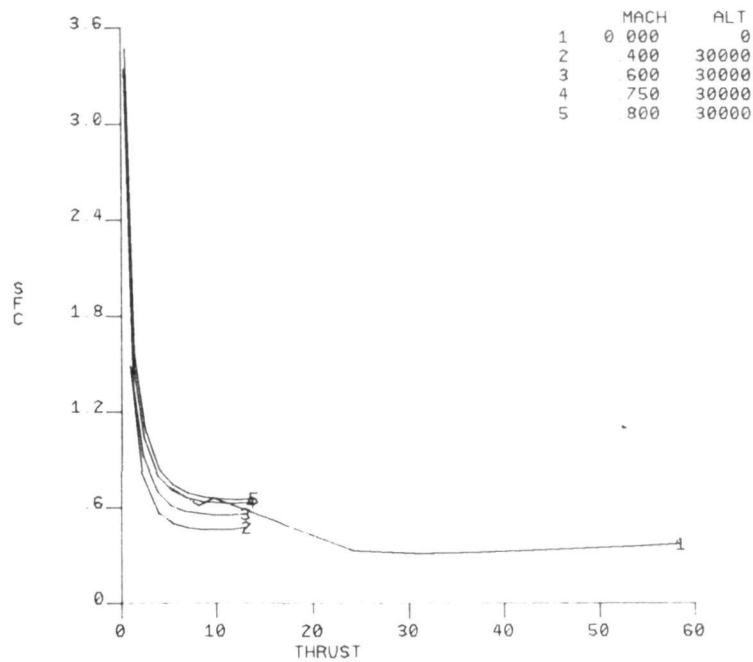


Figure 11.— Throttled engine characteristics.

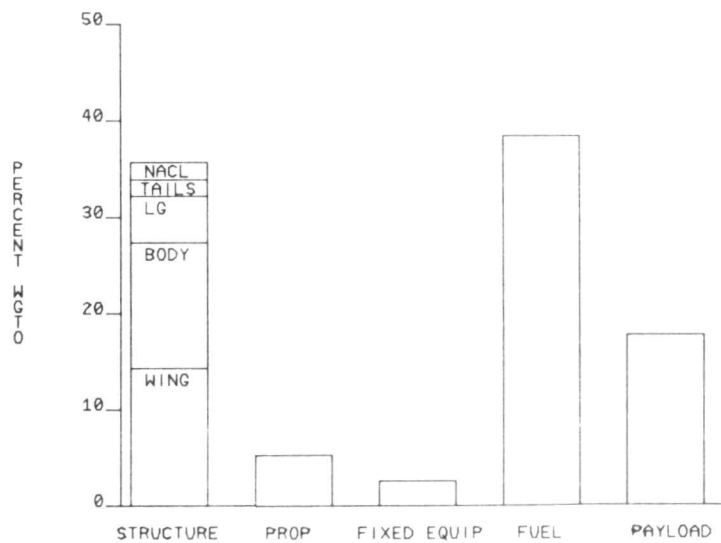


Figure 12.— Aircraft mass distribution.



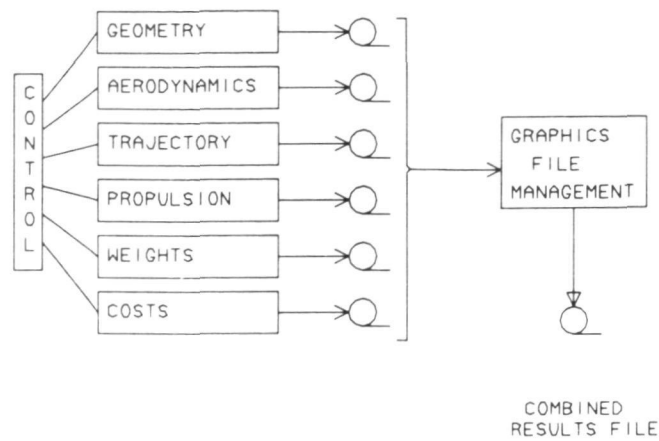


Figure 13.— ACSYNT graphics file management.

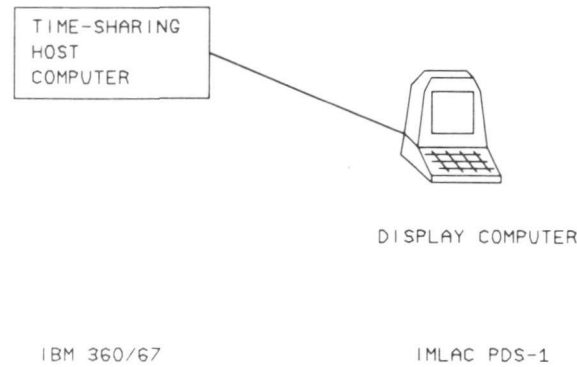


Figure 14.— Original concept of ACSYNT graphics.

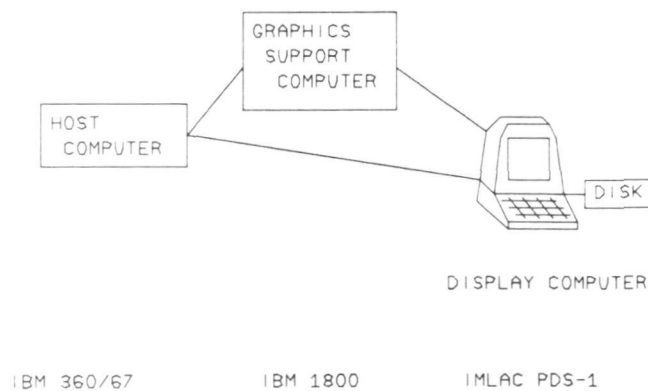


Figure 15.— "Fast-batch" graphics.

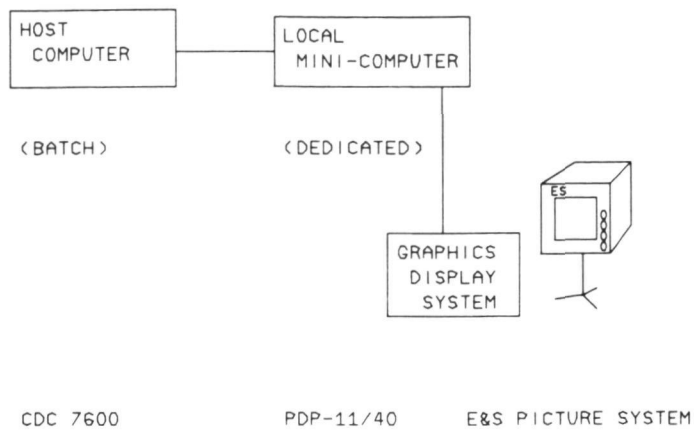


Figure 16.— ACSYNT support processor.

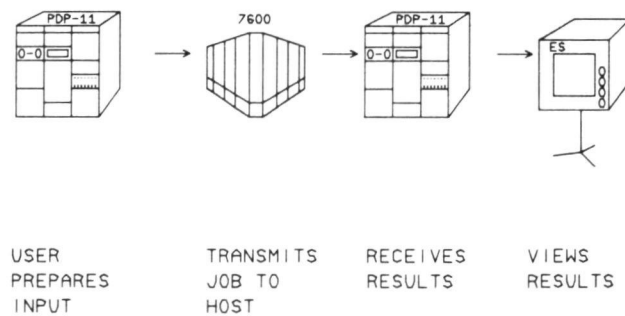


Figure 17.— Operational scenario.

**Page Intentionally Left Blank**

USE OF GRAPHICS IN THE DESIGN OFFICE  
 AT THE MILITARY AIRCRAFT DIVISION  
 OF THE BRITISH AIRCRAFT CORPORATION

By W. A. Coles, B.A.C., Military Aircraft Division, Preston

SUMMARY

This paper describes the CAD/CAM system which has been developed by the Military Aircraft Division of the British Aircraft Corporation, shows the uses to which it has been put, and outlines current development of the system. The system supports batch, time sharing, and fully interactive graphic processing. Engineers using the system may switch between these methods of data processing and problem solving to make the best use of the available resources. It is concluded that the introduction of on-line computing in the form of teletypes, storage tubes, and fully interactive graphics has resulted in large increases in productivity and reduced timescales in the geometric computing, numerical lofting and part programming areas, together with a greater utilisation of the system in the technical departments.

DEVELOPMENT OF THE SYSTEM

The system has been developed from three programs.

1. The BAC Numerical Master Geometry Program (NMG)

This is a language based surface definition, interrogation and machining program. This program, developed by BAC, was introduced into the Design Office during 1965/1966. The inputs and outputs of the program are, in general, finite three dimensional curves and surfaces defining the shape of an aircraft.

2. A.P.T.

This language based,  $2\frac{1}{2}$  - 3 dimensional component definition and machining program, became available to BAC when the IBM APT system was released in 1968. The program has considerable geometric and arithmetical calculation power, but bounded geometry could only be created by cutter motion statements. It is adequate, but not very suitable, for producing component drawing.

This is a fully interactive 2 - 2½ dimensional component definition and machining language. All input to the program is entered using the light pen and keyboard of the IBM 2250. 2 dimensional component drawings can easily be produced.

The starting point for the development of the system was, therefore, two complementary programs, (APT and NMG) which have some overlapping and some incompatible features, but whose only interface was the manual transcription of data, e.g.,

Complementary to the extent that:-

NMG is concerned with aircraft surfaces, whereas APT is concerned with the component geometry - lines - circles, etc.

Overlapping in that:-

Both systems use transformation matrices, interpolated curves, etc.

Incompatible to the extent that:-

- 1) Data derived from the NMG system could not be used in APT and vice-versa, e.g.,

Curves used on APT are similar but not identical to NMG spline curves.

- 2) NMG surfaces could not be machined in APT.

This situation was clearly unacceptable, and it was decided that extensive modifications to the IBM APT programs would have to be made in order to achieve compatibility, and to interface it to the NMG system.

The following facilities were, therefore, added to the APT system:-

- 1) Metric capability included
- 2) Capability to READ and WRITE NMG Files (Access and write, curves, surfaces, and reference systems.)
- 3) Capability to use NMG curves in place of Tabcyls.
- 4) Matrix definitions were changed to be compatible with NMG
- 5) Use of NMG surfaces as APT part, check, or drive surfaces
- 6) Restructured to fit into 140K of core.

The modified APT program which also included extensive modification to improve efficiency was named APT 140 and released in 1971. As expected the combined APT/NMG system provided a powerful geometric computing lofting and N/C machining facility, limited only by the restrictions of batch processing.

Since then the system has been further developed (Fig. 1) to interface with the interactive N/C graphics program, to run semi-interactively under the IBM TSO system using teletypes and remote displays, and to run fully interactively using an IBM 2250.

The interface with the N/C graphics program is via a file which contains component descriptions in the form of bounded geometry and text.

Interactive processing has been achieved by providing the APT and NMG programs with restart facilities, and common input, output, and display files; all of which are under the control of a terminal program. The main functions of the terminal are:-

- 1) Preparation of text input (card images)
- 2) Modifications of text input
- 3) Inspection of text output
- 4) Storage and retrieval of text input
- 5) Graphical preparation of input
- 6) 3 Dimensional picture preparation and manipulation
- 7) Preparation of plots
- 8) Loading program modules.

The introduction of on line processing and graphic displays, together with the 'program independent' terminal facility has provided a powerful impetus to extending the use of the Numerical Master Geometry system to the preparation and verification of geometric information for technical calculations. e.g., Generation of grids for structural analysis and aerodynamic calculations.

#### USE OF THE SYSTEM

The system is extensively used at all stages of design and production as described in the following examples. When considering these examples it is important to remember that for all applications the designer, loftsmen, engineer, part programmer etc., may switch between BATCH, TIME SHARING and FULLY INTERACTIVE methods of data preparation or problem solution.

The method chosen will depend not only on the most 'economical' method of producing the desired result, but on the priority of the job, the departmental loading, computer resources available etc. This flexibility of operation has been found to be of fundamental importance not only because particular resources may be withdrawn due to breakdowns, but because relative priorities change with time, and jobs started in batch may be completed interactively and vice versa.

1) Lines Definition Fig. 2, 3, 4, 5

The project designer prepares a scheme, which is passed to the linesman who uses the NMG system to convert the scheme into the final aircraft lines which are stored as a surface on the NMG Master File. The scheme will normally consist of a mixture of dimensioned and drawn data, and the linesman will use a mixture of calculation and drawing to establish the set of sections which are submitted to the NMG program for conversion into a surface. The process of achieving a fair set of lines is iterative, the designer will examine definition data, water lines and level lines between iterations and modify the definition sections until a smooth set of lines has been obtained (Fig. 3 and 4). When he is satisfied with the surface it will be stored on the master file, and a surface definition drawing will be issued. (Fig. 5).

Surfaces representing the whole aircraft (Fig. 6) including internal and external surfaces (Fig. 7) are generated and stored on the master tape.

All information in the master tape is stored in the basic aircraft reference system.

Surfaces on the master tape are now available for general use. Typical uses for the surfaces are:-

- 2) Nests of sections (Fig. 8)
- 3) Obscuration plots
- 4) Clearance problems (Fig. 9)
- 5) Machining wind tunnel models (Fig. 10)
- 6) Machining form tools (Figs. 11 and 13)
- 7) Generation of new sections which form the outside shape of components (Fig. 12, 13)

Here we see a linesman typing a request for a new section to be generated and plotted. The teletype will only return a listing but a graphic display (Fig. 13) will return a picture which can be checked for errors prior to plotting on the Gerber Draughting machine. Input to the Gerber Draughting Machine is generated by typing the instruction "PLOT GERBER SCALE n" which produces a Gerber input tape of the picture displayed on the screen.

#### 8) Definition and Machining of Large N/C Components

Large N/C components are numerically lofted. The outside shape is taken from the master file, and the details of the internal geometry are added using the component definition languages. The manufacturing drawing issued by the loft carries a reference to the component geometry and the identifiers of the geometric variables. (Fig. 14, 15). To produce the N/C machining tape the part programmer calls up the component geometry and adds his own cutter statements. The cutter path can be checked by superimposing it on the component geometry before producing the final machined component. (Fig. 16).

Fig. 15 shows a typical drawing and component produced using the APT program. Figs. 17 and 18 show a similar component being defined and machined using interactive graphics. Interactive graphics is being increasingly used for N/C lofting and N/C part programming, and the APT program is being modified to extend interactive techniques to three dimensional components (Figs. 19 and 20).

#### 9) Geometric Data for Aerodynamic and Structural Analysis

The generation of data for automatic transfer to aerodynamic and structural analysis programs presents a particularly challenging problem which is receiving increased attention. A frequently encountered task is the generation of a grid of points forming a set of panels.

At face value this would appear to be a simple task, but considerable difficulties and complexities are introduced by:-

- 1) The variety of methods used to specify point locations and panel sizes. This problem is particularly severe for aerodynamic calculations where panel topology is directly linked to aerodynamic theory and must frequently be derived from surface properties such as local chord, span, aspect ratio etc., which are no longer held explicitly in the master model.



- 2) Aerodynamic and structural grids are usually formed on a group of surfaces some of which may overlap (Fig. 21 shows a simple case). A cut through such a group of surfaces results in a set of curves which may easily be drawn, but which are not suitably ordered for subsequent processing to form a grid.
- 3) Only part of the analysis data can be derived from the master model. Local modification may be required, and the data will have to be combined with manually prepared data.
- 4) The format of the output is dependent on the program with which it is to be used.

In the batch processing environment these problems proved sufficiently severe to restrict us to interfaces based on the manual transcription of data. The advent of on line devices has, however, renewed the hope that the problems can be overcome, and Figs. 21 and 22 show a promising method suited to graphic devices which is being investigated.

The grid of Fig. 21 is produced by finding all points of intersection of the lines GL1 etc., with the nest of sections, and it is seen that the picture must be modified in the region of GL4 in order to define a unique set of panel points. On line graphic devices are particularly suitable for generating, checking, and modifying pictures of this type which can be subsequently batch processed to produce panel data. Similarly the structural analysis grid of Fig. 22 can easily be drawn and checked using a graphic device, and be subsequently batch processed to produce point and panel data.

The output data for both grids is written in the appropriate format and stored as card images in the on line 'card' file. From there it can easily be modified and combined with manually prepared data using terminal facilities.

## CONCLUSIONS

The addition of on line teletype and graphic facilities to augment the previous batch processing system has resulted in increases in productivity of the order of 50% to 100% in the geometric computing, numerical lofting, and NC programming areas. Timescales for high priority jobs can be reduced even more dramatically. The main factor contributing to this improvement is that the majority of delays encountered in running a fragmented system have been eliminated, e.g., delays associated with conventional data processing:-

Manual Transcription of data

Waiting for cards

Waiting for loading into the computer

Waiting for listings

Waiting for plotting

are avoided.

Streamlined data processing is particularly important when a number of consecutive small jobs have to be run to establish the data for a more substantial task.

Further benefits from the system are that:-

N/C components produced interactively are more often correct first time than those produced using batch processing. On line computing has truly augmented the batch processing system. Graphical displays can be used to check data before embarking on expensive batch processing runs, and users no longer feel obliged to run large jobs to establish as much data as possible overnight in order to avoid a day's delay if anything has been forgotten. The Gerber Draughting machine is released from the task of producing small scale check plots.

The concept of a program and device independent text based system, with common input, output and display files, operating from card checks, typed input or graphically prepared data is proving sound, and is providing a powerful impetus to the use of graphic data processing in all technical departments.

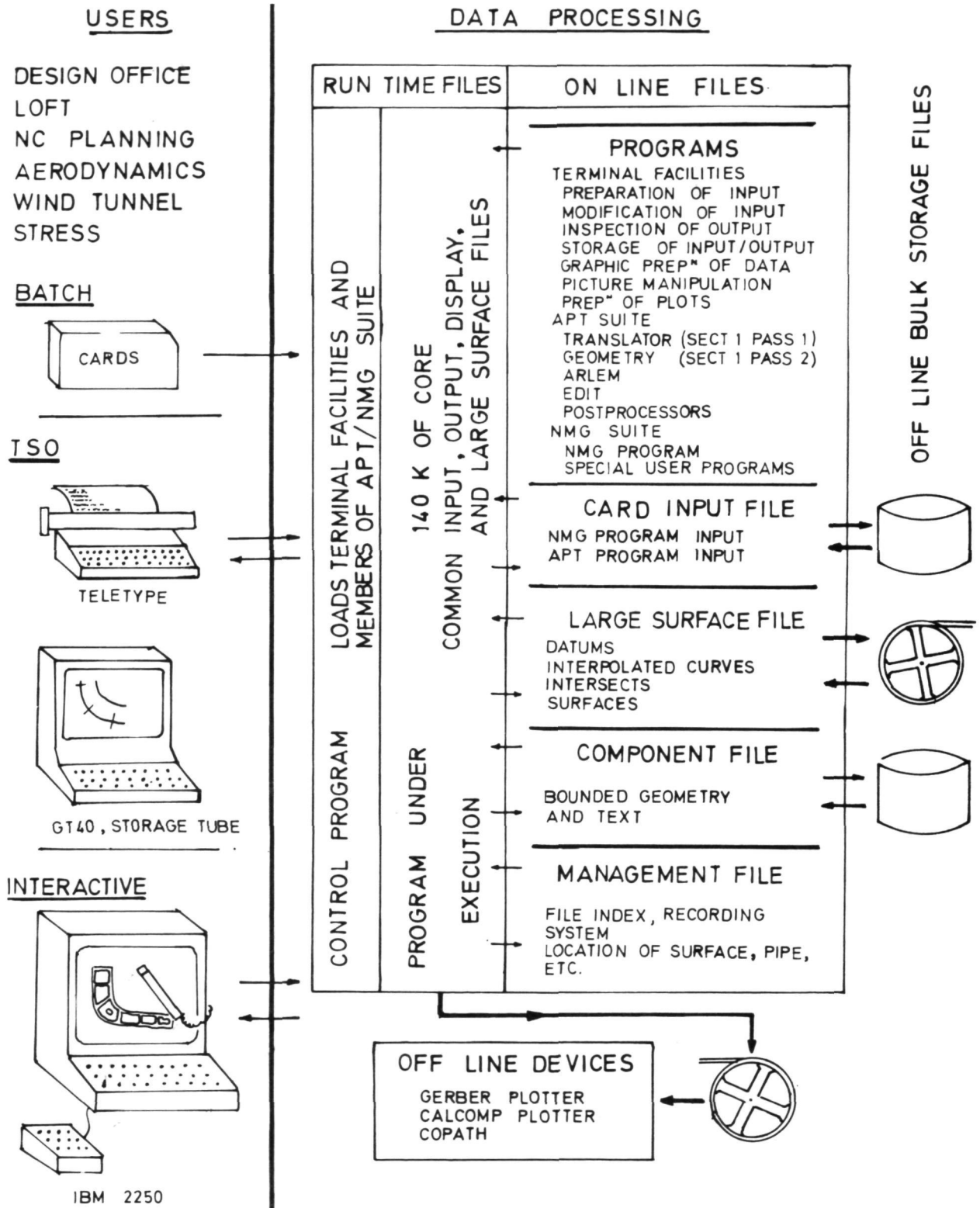


Figure 1.- The CAD/CAM system.

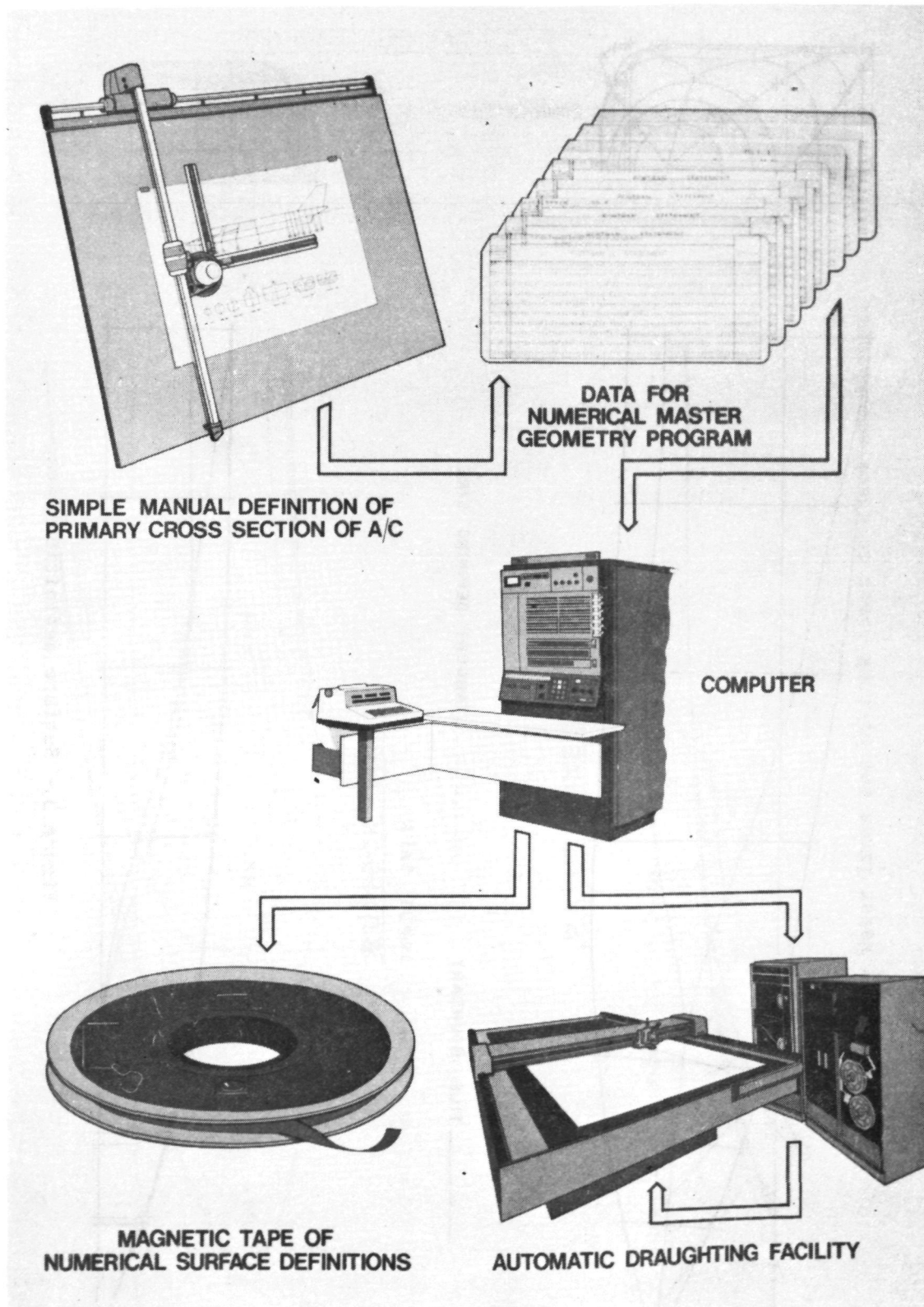
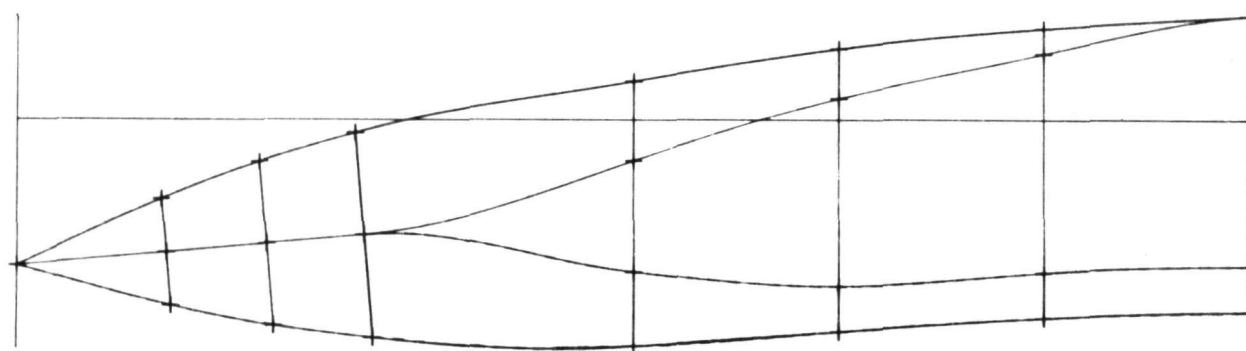


Figure 2.- The Numerical Master Geometry system.



——— TILE BOUNDARY      - - - - - TANGENT DEFINING LINE  
 +      INPUT DATA

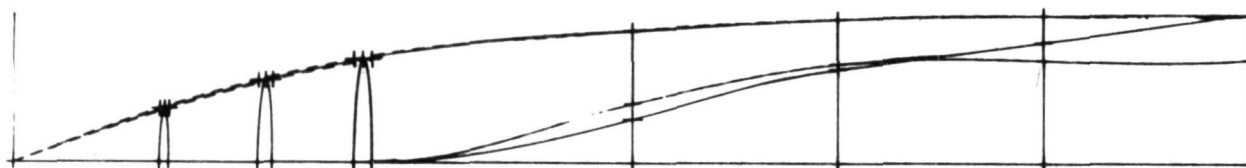
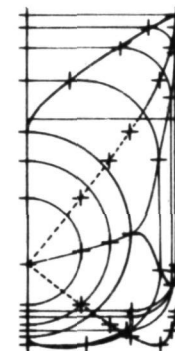
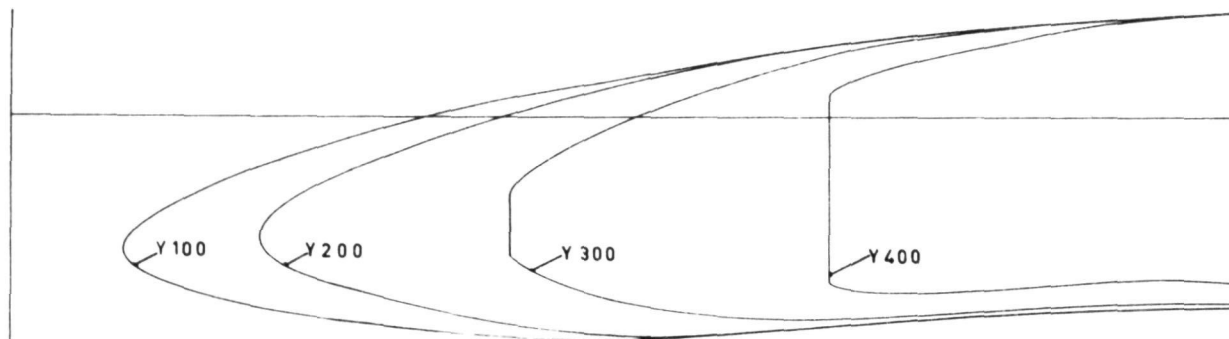


Figure 3.- Surface definition data.



BUTTOCK LINES.



WATER LINES.

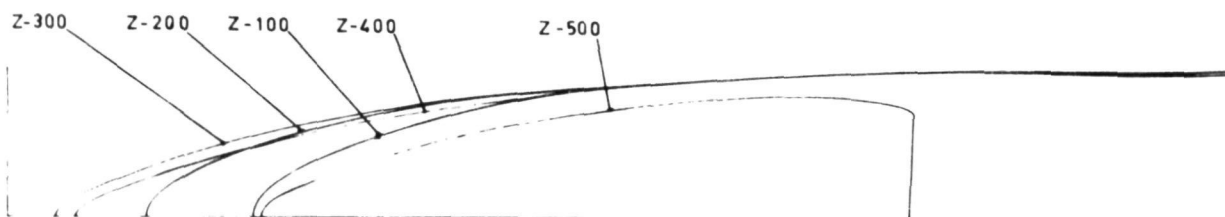
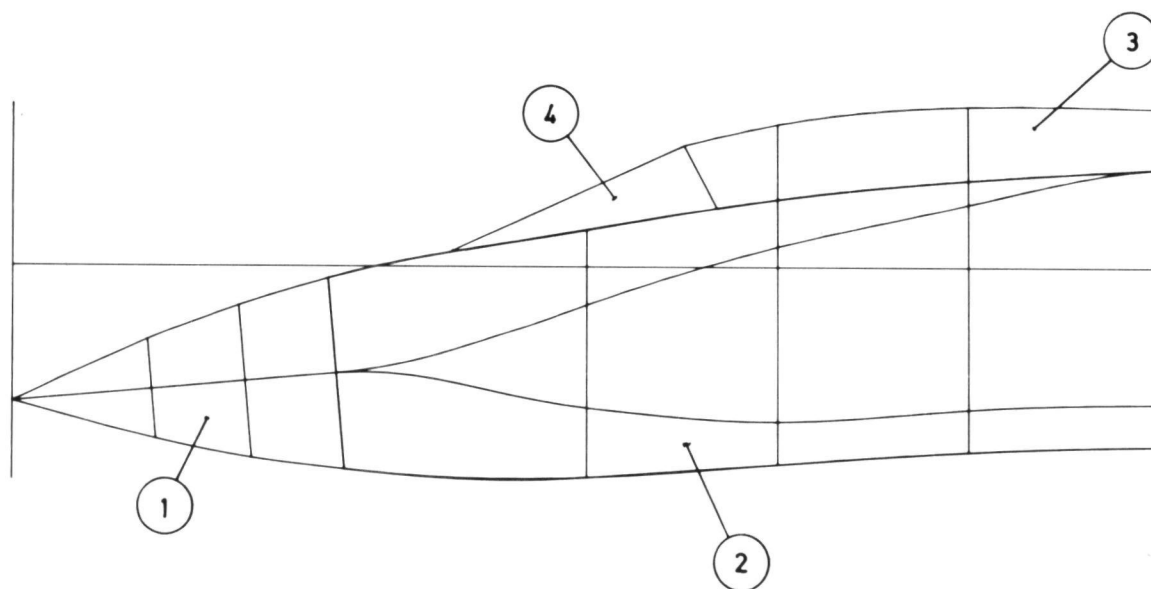
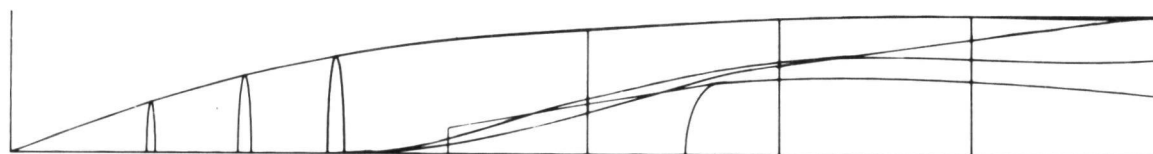


Figure 4.- Water lines and buttock lines to check smoothness.

NOTE

EACH SURFACE IS DRAWN REFERENCING U AND V LINES.



4	D/20/S 40
3	D/20/S 7
2	D/20/S 2
1	D/20/S 1
Nº	ITEM NAME

FILE NAME\_DEMOMAST

Figure 5.- Surface definition drawing referencing master files.

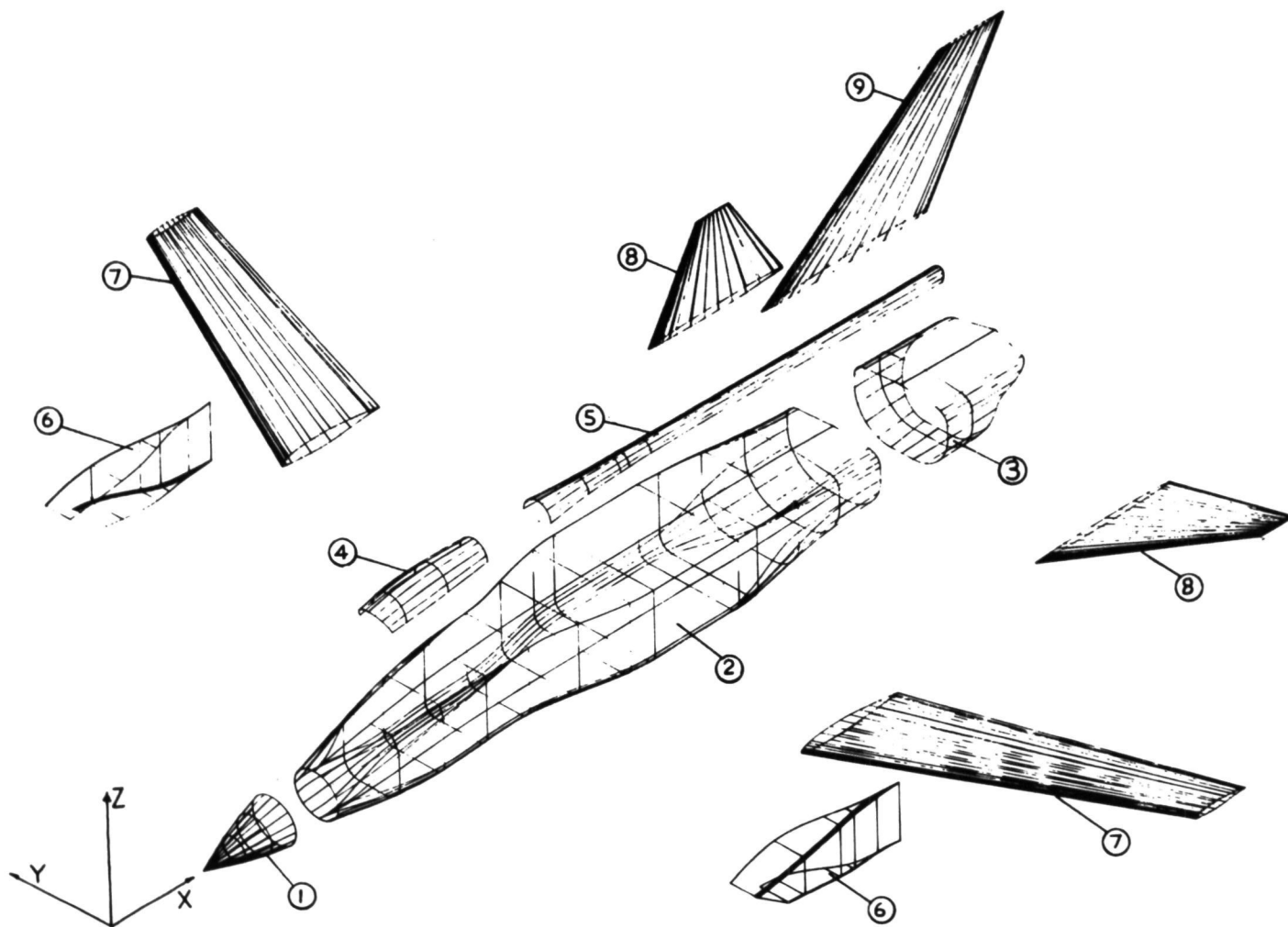
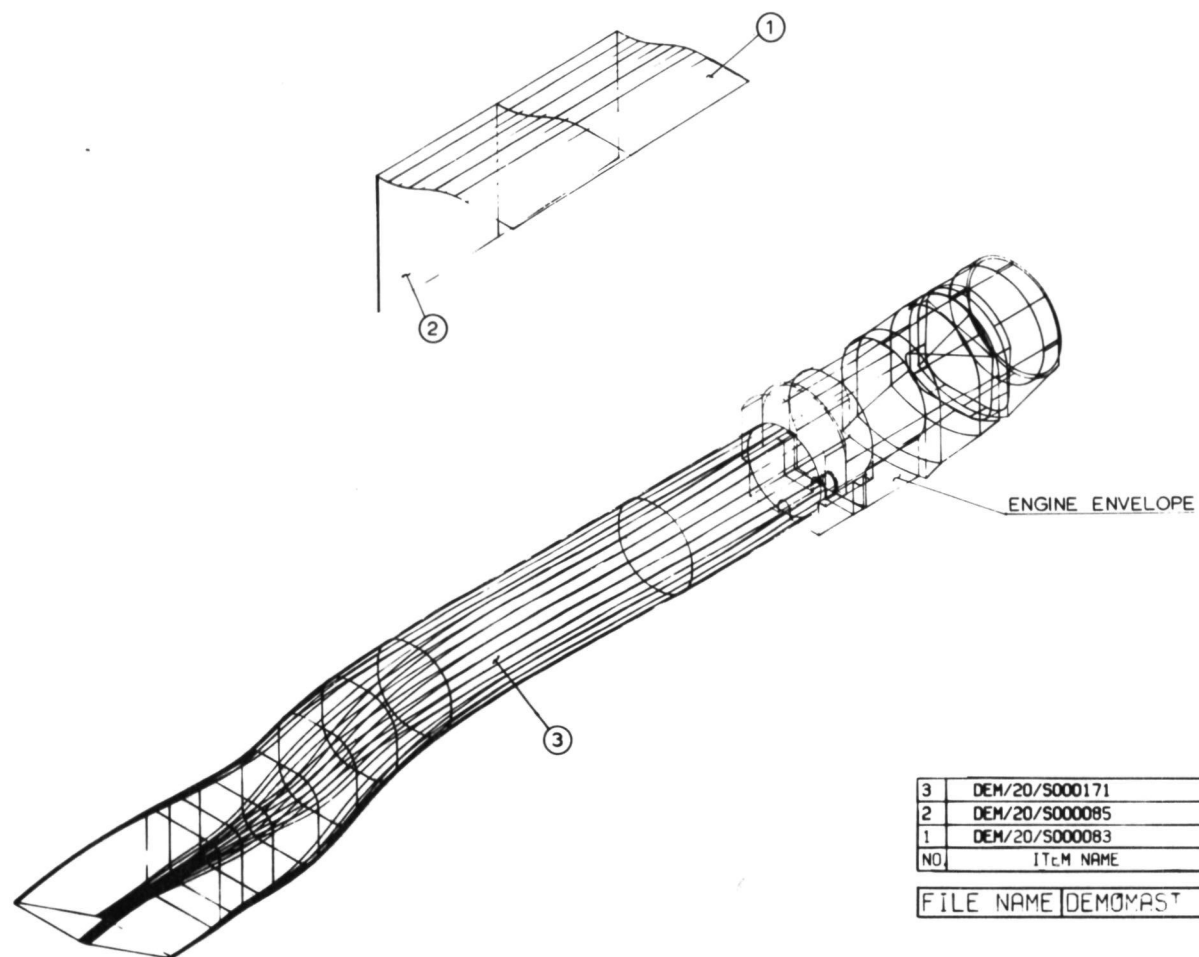


Figure 6.- Isometric view of typical aircraft surfaces.





3	DEM/20/S000171
2	DEM/20/S000085
1	DEM/20/S000083
NO	ITEM NAME
FILE NAME DEMOMAST	

Figure 7.- Surfaces defining duct, engine bay and engine envelope.

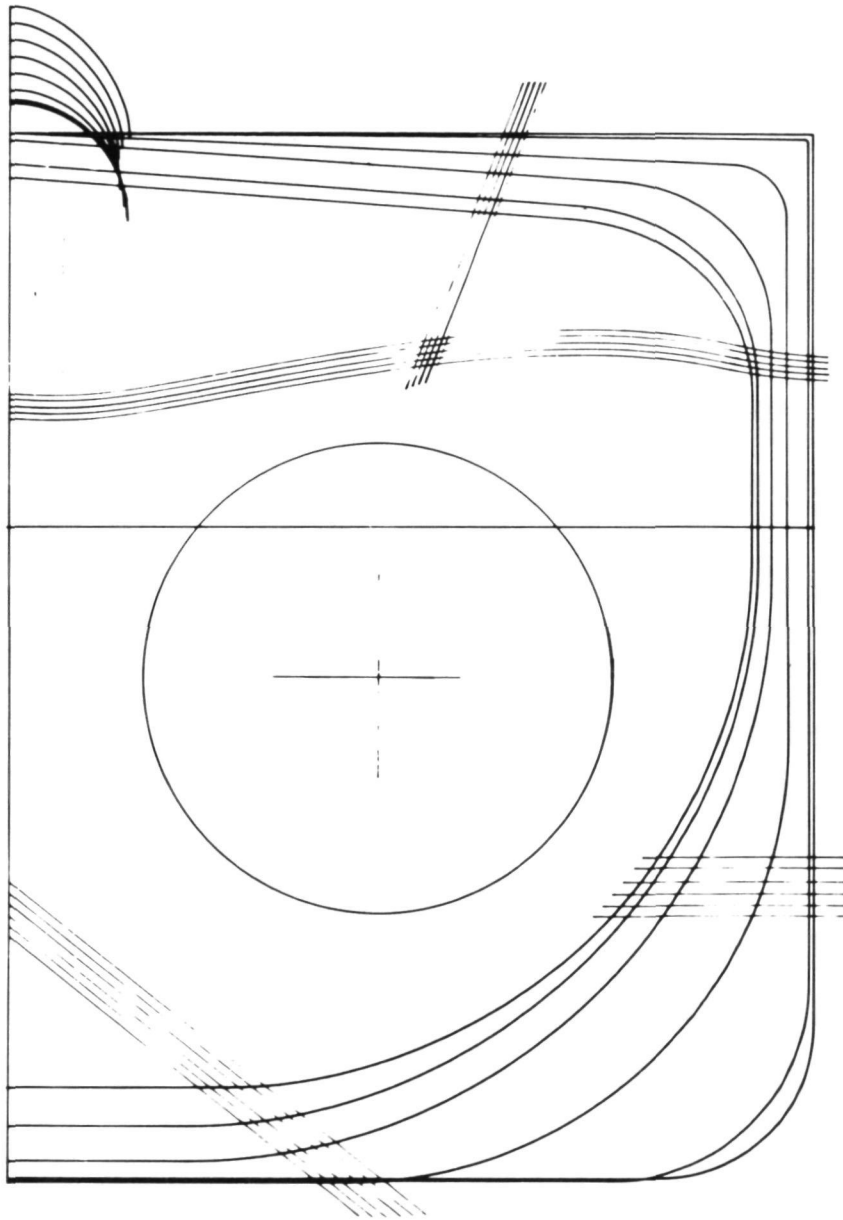


Figure 8.- Nest of sections showing location of internal structure and engine tunnel.

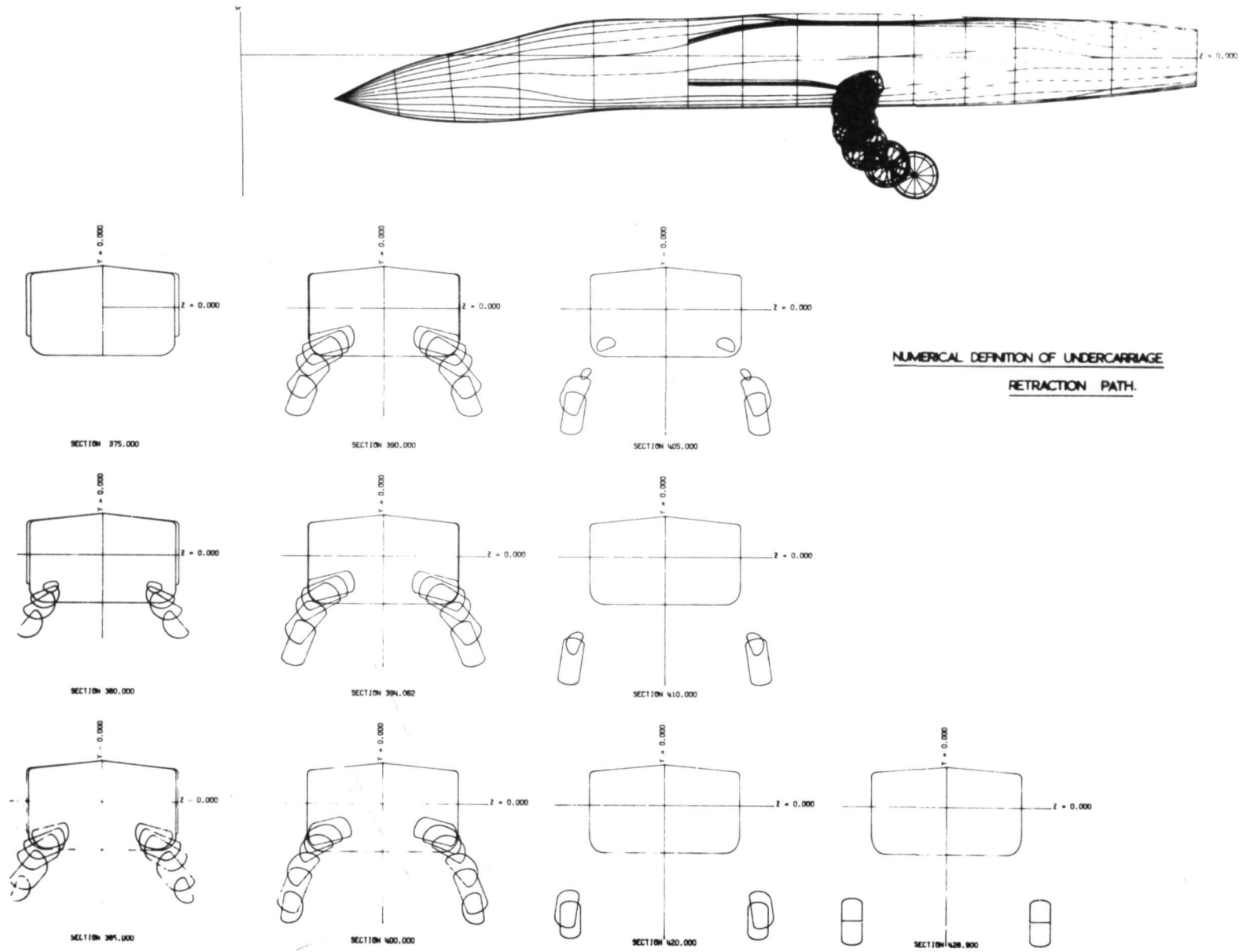


Figure 9.- Clearance checks.

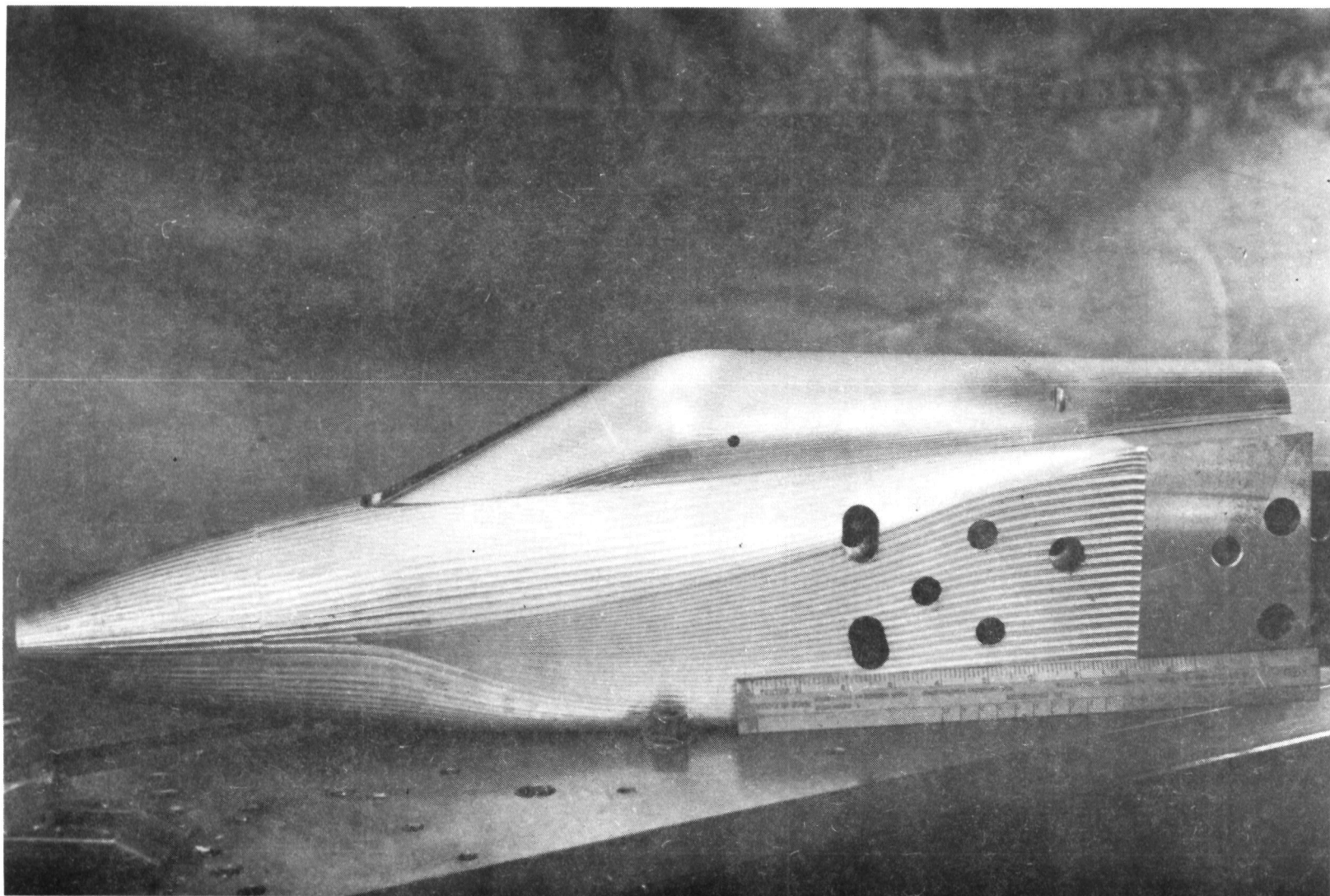


Figure 10.- N/C produced wind tunnel model.

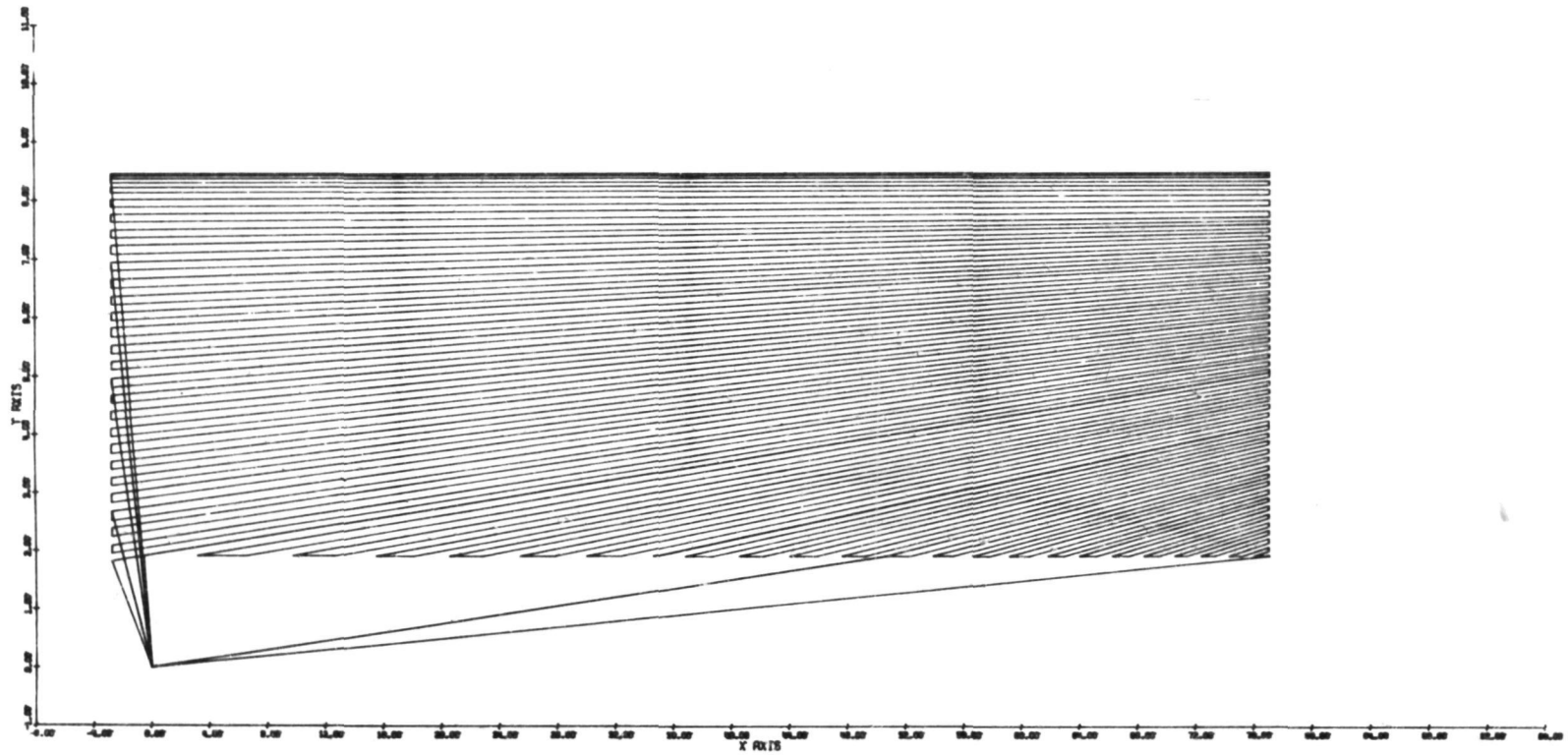


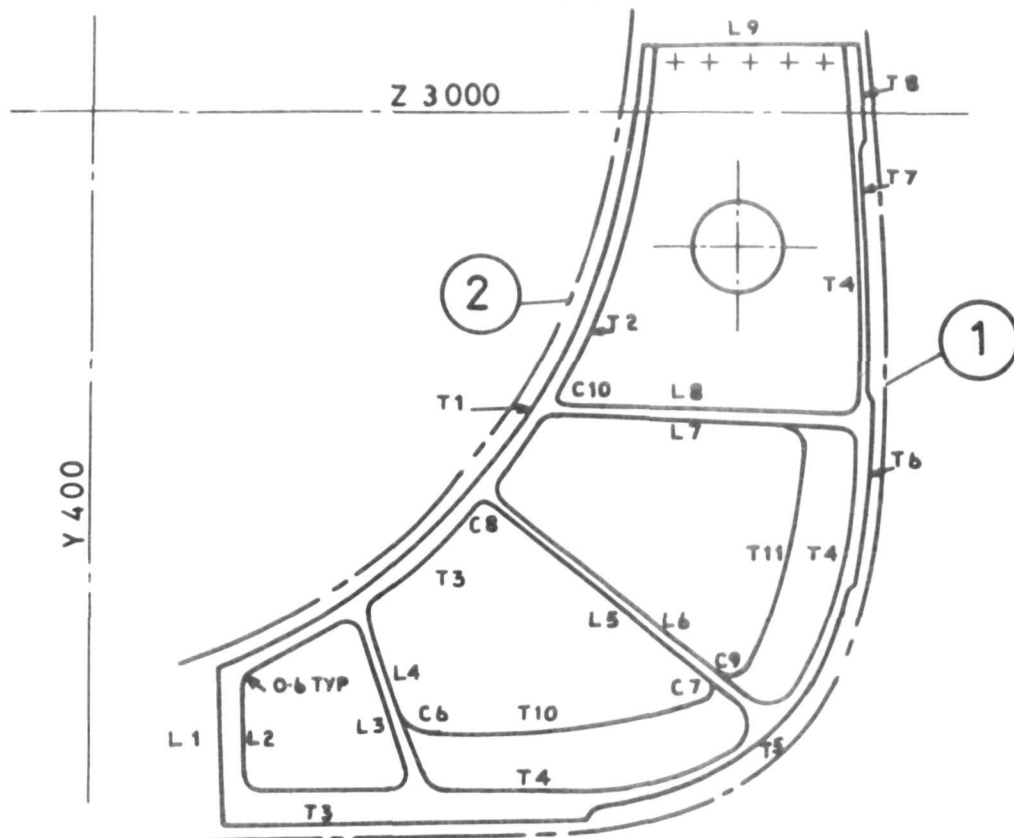
Figure 11.- Cutter path for N/C machining a form block.



Figure 12.- Time sharing teletype terminal.



Figure 13.- Time sharing graphics terminal.



GEOM AO 8400E6	
NC FILE	
1	P/20/F000682/X4028
2	P/20/F000674/X4028
FILE NAME :- PANAMAST	

### MACHINED FRAME

Figure 14.- Numerically lofted N/C machining drawing.



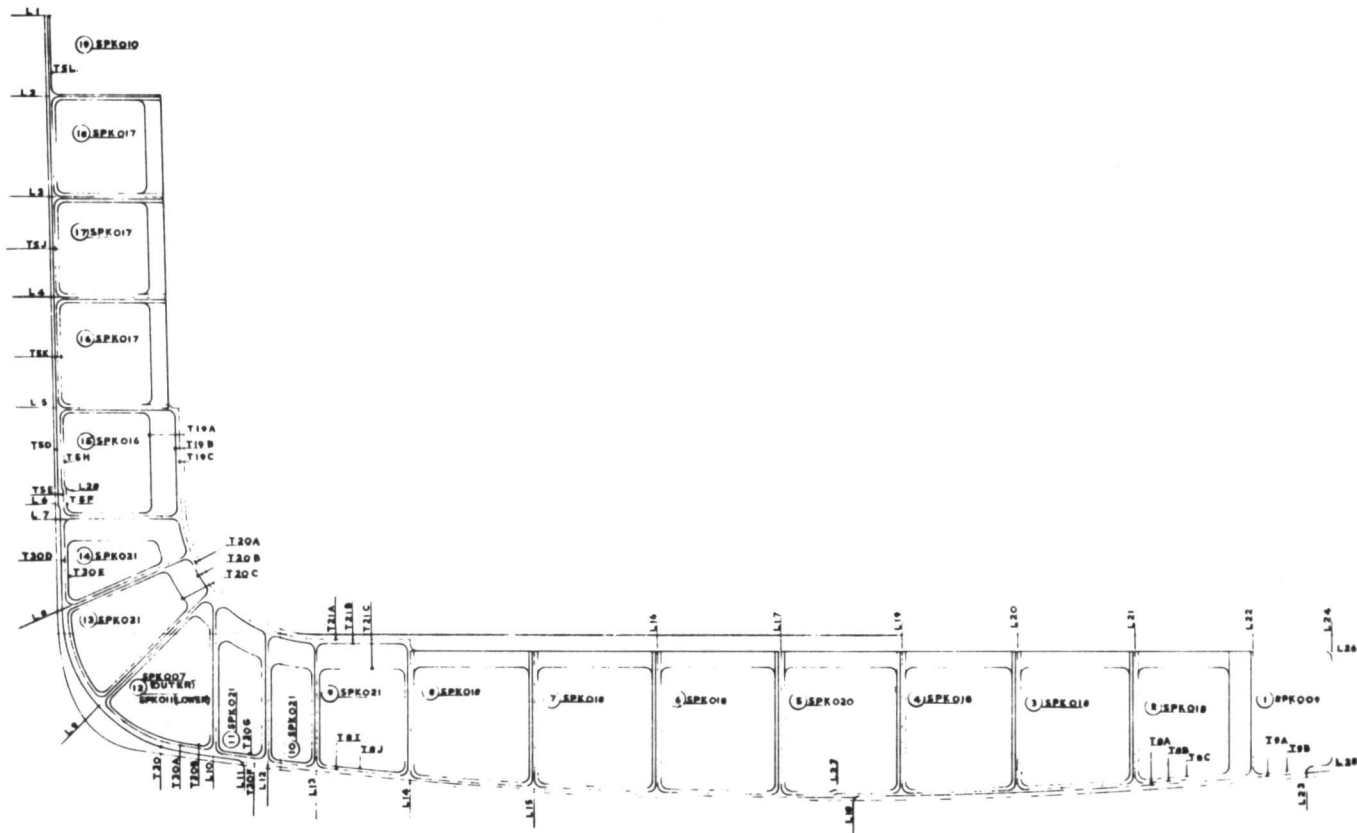


Figure 15.- Concorde intake frame: design produce a numerical definition and drawing which are passed to production.

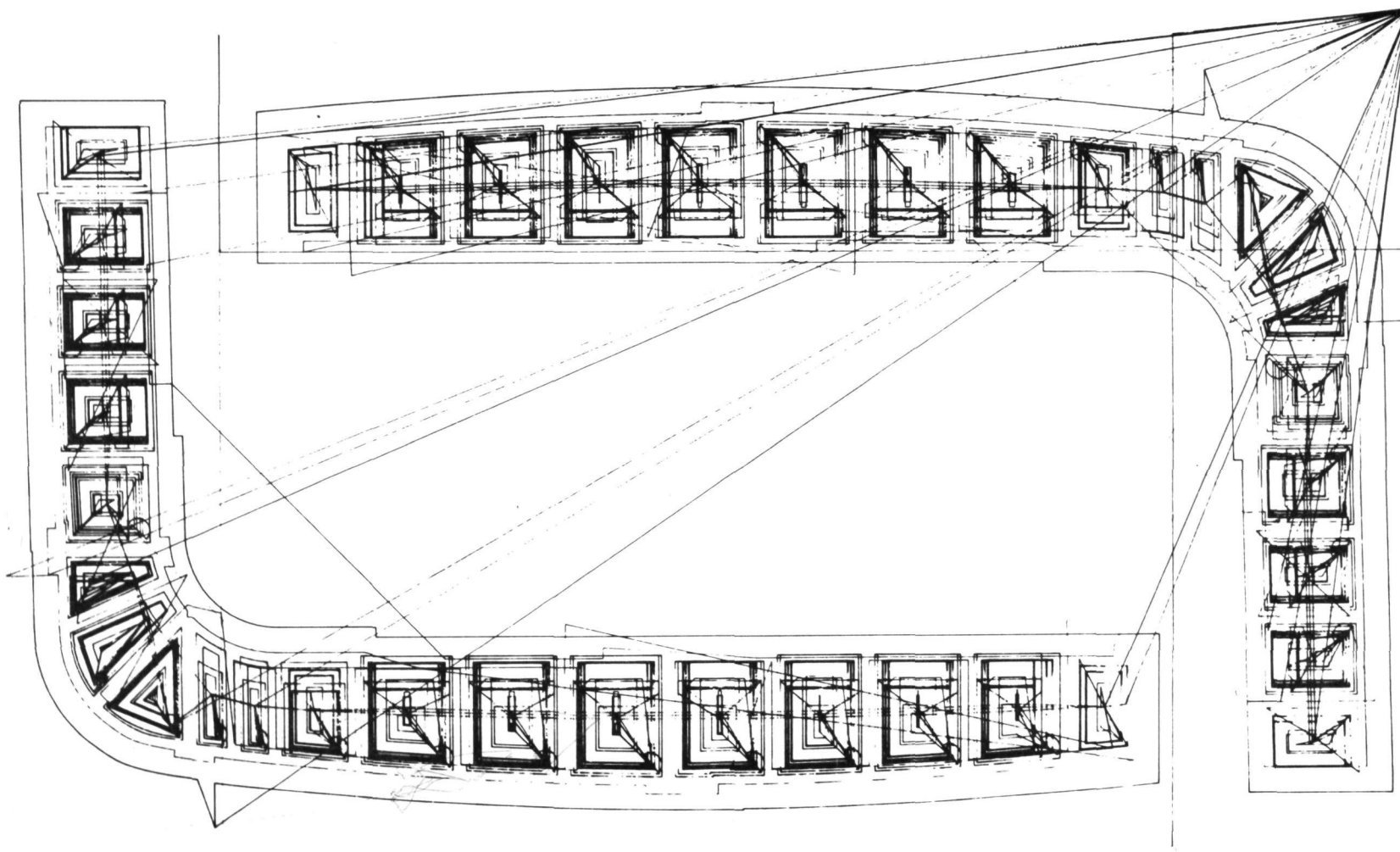


Figure 16.- Concorde intake frame: production plan the machine layout and cutter path.

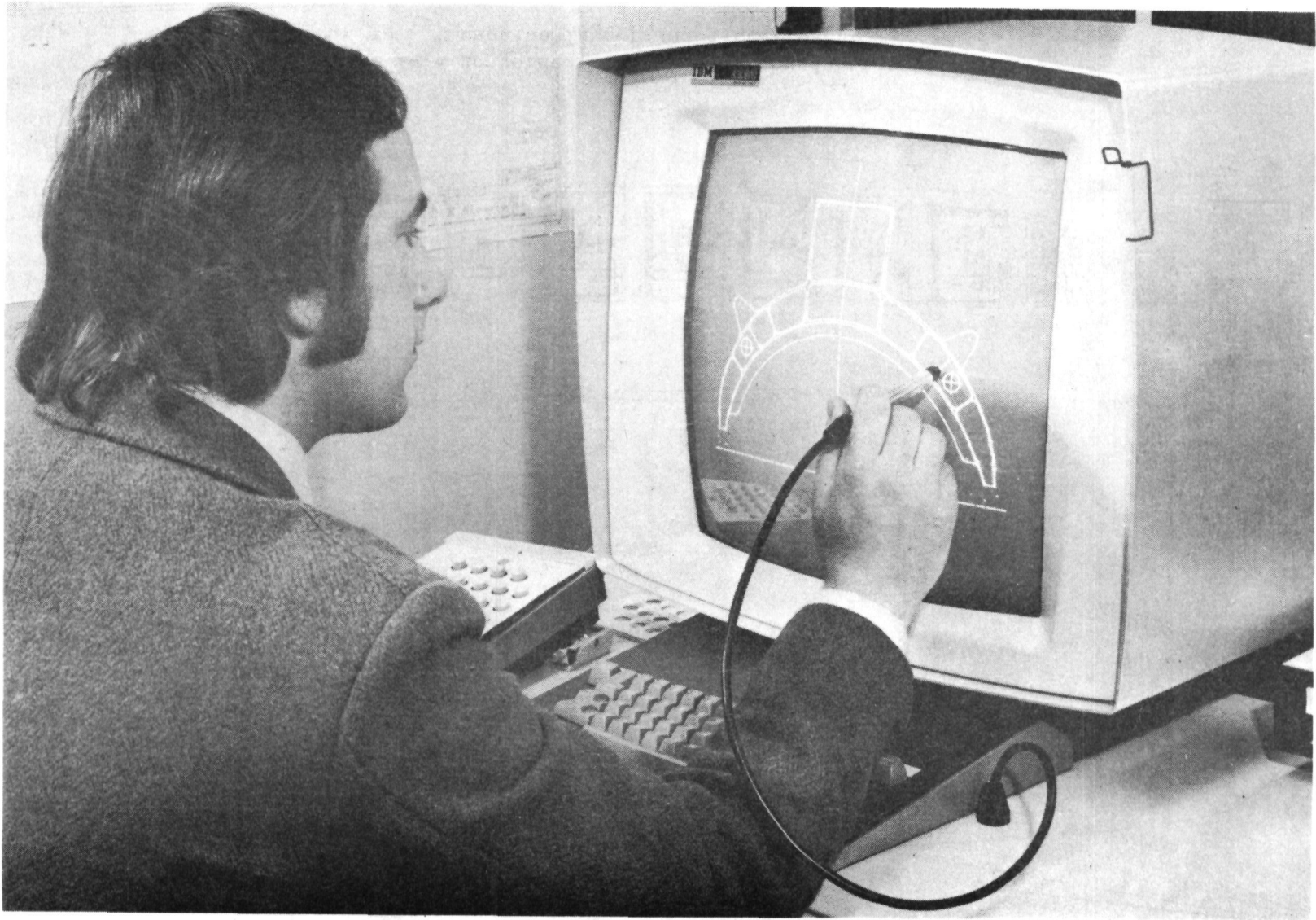


Figure 17.- Numerical lofting using interactive graphics.

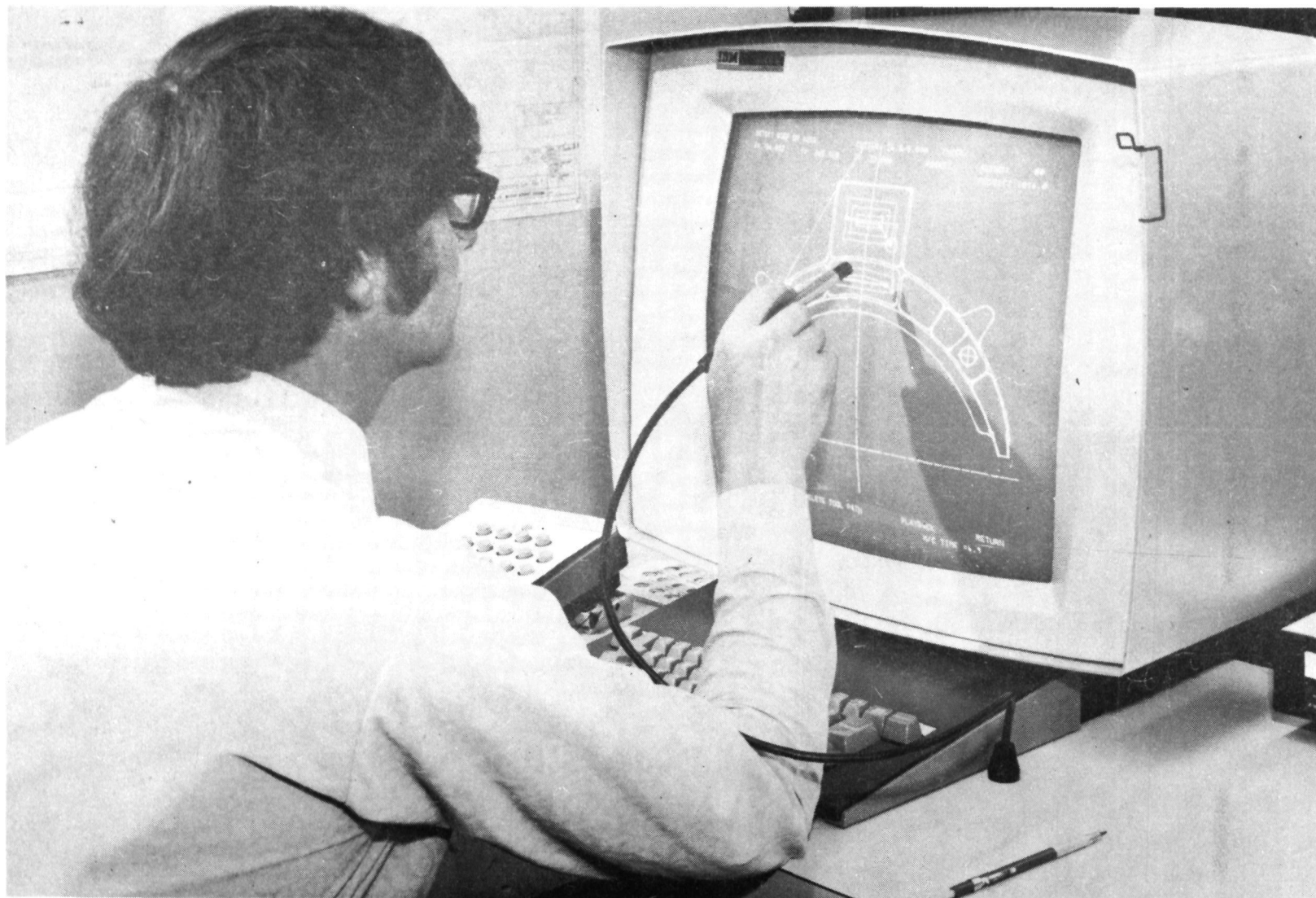
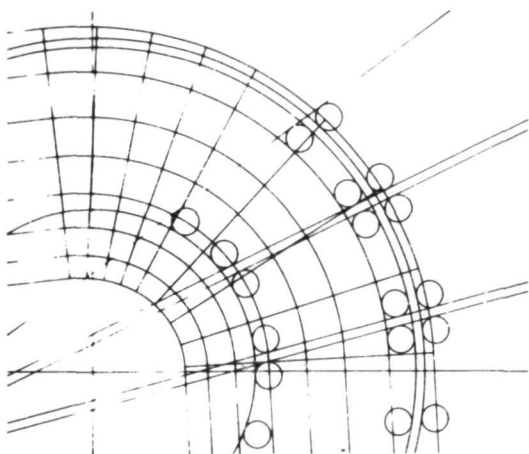


Figure 18.- Numerical machining using interactive graphics.

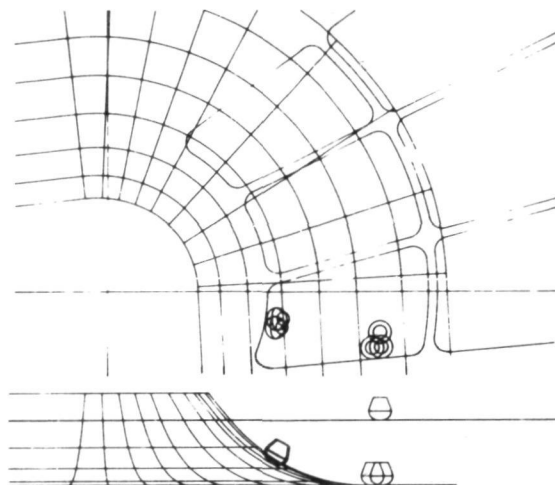


Figure 19.- Interactive APT using IBM 2250.

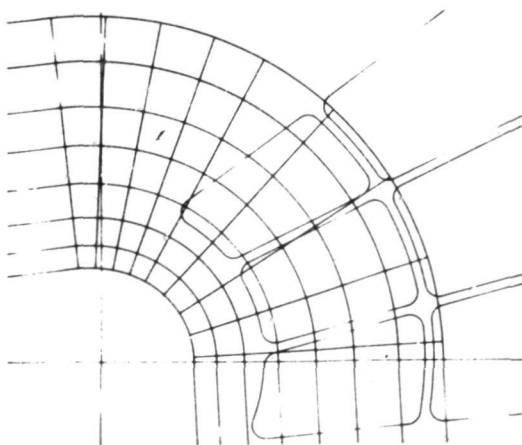
## UNBOUNDED GEOMETRY



## MACHINING A POCKET



## BOUNDED GEOMETRY



## CUTTERS

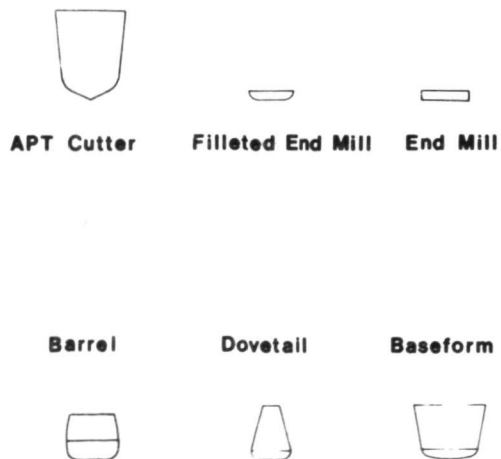
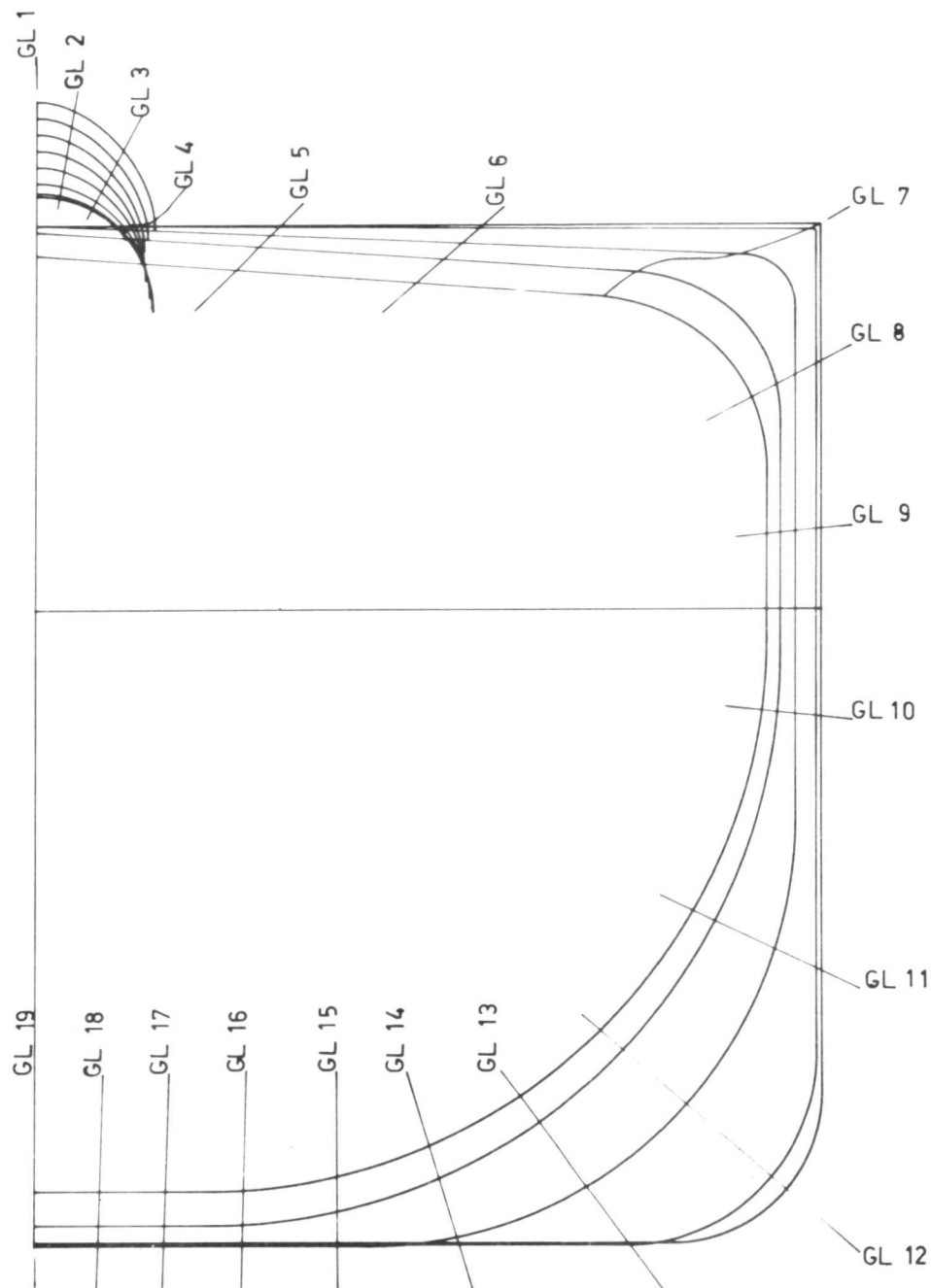


Figure 20.- Producing bounded geometry with interactive APT.



GRID LINES (GL...) DEFINED BY  
BOUNDED GEOMETRY PANEL POINTS  
FOUND WHERE GRID LINES INTERSECT  
THE NEXT OF SECTIONS.

Figure 21.- Panelling for aerodynamic calculations.



GRID LINES DEFINED BY BOUNDED GEOMETRY. STRUCTURAL ANALYSIS

NODES FOUND BY PROJECTING GRID INTERSECT POINTS ON TO SURFACES.

STRUCTURAL ANALYSIS ELEMENT DATA FOUND FROM  
NODE CONNECTIVITY DATA

e.g. GS5 INTERSECTS GR17 AT NODE 15

GS5 INTERSECTS GR 9 AT NODE 15

etc.

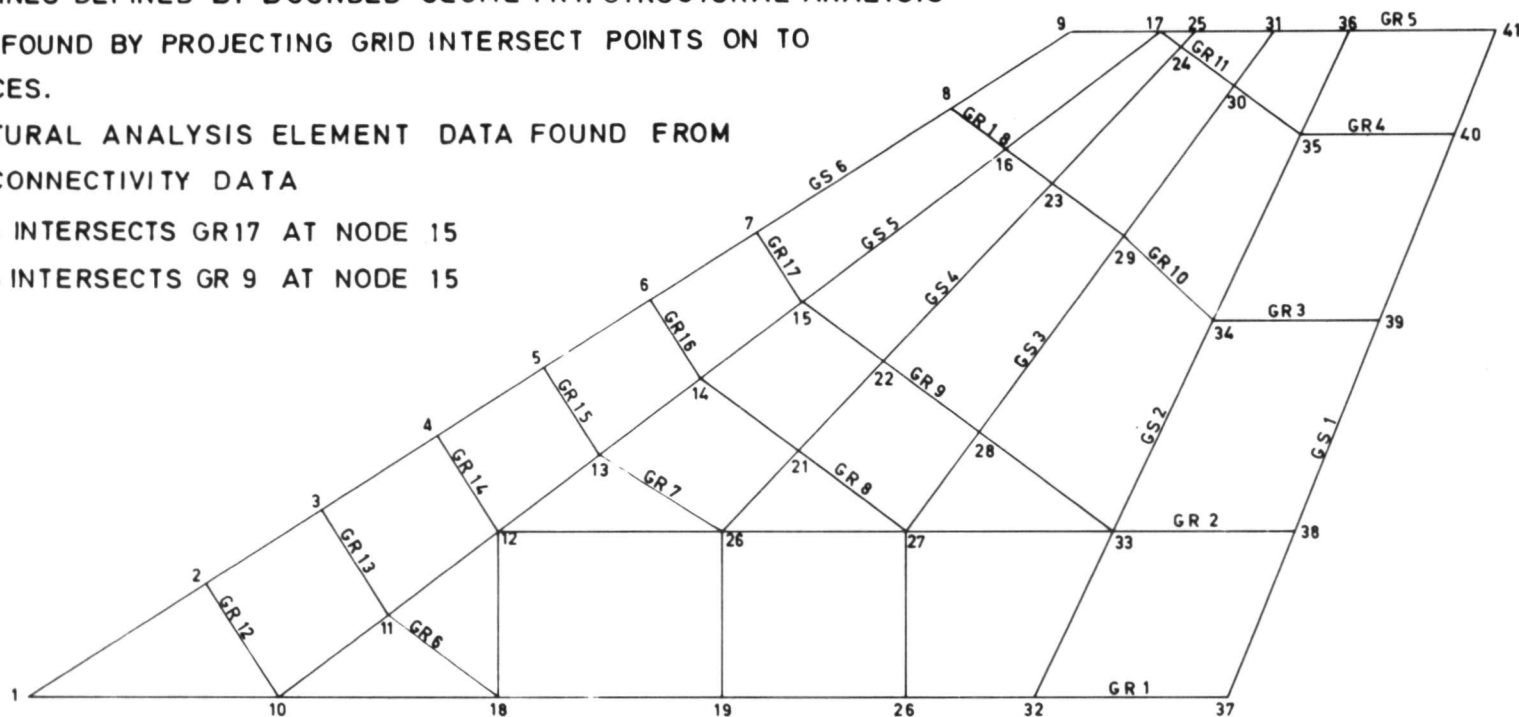


Figure 22.- Structural analysis grid.



**Page Intentionally Left Blank**

# INTERACTIVE COMPUTER GRAPHICS SYSTEM FOR STRUCTURAL

## SIZING AND ANALYSIS OF AIRCRAFT STRUCTURES

By Dror Bendavid, Aharon Pipano, Alex Raibstein  
and Emil Somekh

Israel Aircraft Industries Ltd.

12

### SUMMARY

A computerized system for preliminary sizing and analysis of aircraft wing and fuselage structures is described. The system is based upon repeated application of analytical program modules, which are interactively interfaced and sequence-controlled during the iterative design process with the aid of design-oriented graphics software modules. The entire process is initiated and controlled via low-cost interactive graphics terminals driven by a remote computer in a time-sharing mode.

### INTRODUCTION

The macro design process of flight vehicle structures is typified by a hierarchy of intra and inter-disciplinary iterative procedures involving numerous design modifications and repeated execution of sophisticated computerized analyses. Some of the major disciplines involved in this process are those dealing with geometry, aerodynamics, weights, loads, structural design, stress and aeroelasticity.

One of the major bottlenecks and common sources of errors, delays, and frustrations in the iterative structural design process is associated with the preparation of massive geometry-oriented data required as input to the computerized analyses, and the subsequent reduction of the numerical output data into meaningful decision-worthy form. Indeed, the advent of the digital computer, while permitting for the solution of hitherto unsolved problems, also led to the inadvertent creation of a new breed of engineer. This unfortunate and misdirected creature, commonly referred to as an "I/O Engineer", may be glimpsed running to and fro carrying heavily laden boxes of input cards and output stacks instead of concentrating on his main tasks:

- Creation of a technically acceptable discretized math model generally required for the response solution of complex structures.
- Evaluation of the analysis results and the application of engineering judgement and creativity in modifying the design to better meet the required criteria.

Ideally, the ultimate objective in design should not be the analysis of a given

configuration, but rather the systematic synthesis of a design, starting from a set of given requirements, and satisfying the specified design criteria. However, since true synthesis is not as yet a viable technology, near optimum designs can be achieved only through iterative analysis. Thus, any truly successful computerized design system should provide the engineer with the capability of performing analyses, evaluations, and design modifications in rapid interactive succession, without having to preoccupy himself with the laborious and time-consuming chore of massive geometric I/O preparation and data reduction - an activity which dulls his engineering creativity and restrains him from performing his true functions as an engineer.

In attempting to meet these objectives, IAI is developing ISSAS (Interactive Structural Sizing and Analysis System), which is a modular and highly flexible computerized system for preliminary sizing and design of flight-vehicle wing and fuselage structures. The system is based upon the use of analytical program modules, which may be interactively interfaced and sequence-controlled during the iterative design process through the use of low-cost interactive graphics terminals driven by a large computer in a time-sharing mode.

In developing the ISSAS concept, IAI has followed many of the guidelines layed down in similar computerized structural analysis systems developed over the past five years (references 1-6). However, in ISSAS, major emphasis was placed on enhancing the interactive design capability while at the same time retaining the computing power of the analytical modules. Another major feature is the extensive use of low-cost time-sharing ICG terminals, both for program-control and as an aid to I/O generation, reduction, and interfacing.

#### GENERAL SYSTEM DESCRIPTION

The system consists of six major analytical modules interfaced with six ICG (Interactive Computer Graphics) modules, as depicted in figure 1. The analytical modules are:

- AERO: Aerodynamic analysis module.
- WEIGHTS: Mass properties analysis module.
- LOADS: Intergrated loads analysis module.
- FESA: Structural analysis and automated design module.
- FAMOS: Frequencies and mode shapes module.
- AEROLAS: Aeroelastic analysis module.

The interactive graphics modules are:

- AEROMOD: AEROdynamic MODEling module.
- AEROPOST: AEROdynamic POST-processing module.
- ISLADE: Interactive Structural LAYout and DEsign module.
- SMOG: Structural Modeling Oriented Graphics module.
- GRASP: Graphics Augmented Structural Post-processing module.
- MODIS: Mode shape DISplay module.

These ICG Modules provide the following capabilities:

- Automated/ICG generation and visual checkout of 3-d geometric discretization data required for aerodynamic analysis
- ICG visualization of aerodynamic loads and flow parameters
- ICG design, layout and modification of primary lifting surface and fuselage structures
- Automated/ICG generation, visual checkout and modification of 3-d structural analysis models
- ICG visualization of structural strength response parameters
- ICG visualization of vibration mode shapes

The system is structured such that an engineer working at the graphics console can supply the system with data from the central data base and execute one or more of the analytic or ICG modules in any technically feasible sequence he may choose. The engineer may at any time discontinue the process and restart it later at the previous termination point. Figure 2 depicts a typical ISSAS sequence flow for the design of a lifting surface structure.

Permanent records of alphanumeric and graphic information may be obtained either on-line using a hard copy device or off-line using a line printer or large drum plotter.

## SYSTEM MODULE DESCRIPTION

### Geometry Data Base

The ISSAS geometry data bank, which is stored on-line, contains the external surface definition of the entire aircraft categorized in terms of its major assemblies and sub-assemblies, such as wing, flap, aileron, nose, cockpit, aft-fuselage, tail, etc. Each such item may be interrogated separately or in fused form as to a variety of geometrical parameters. These are presented to the engineer interactively either in alphanumeric form and/or as an on-line picture which may be rotated in space, translated, zoomed, windowed or otherwise manipulated as required. The digital information describing the geometry may be easily interfaced as input data to analytic or ICG modules of the system.

### AEROMOD - The Aerodynamic Modeling Module

AEROMOD is an ICG module that permits the engineer to rapidly generate aerodynamic panel meshes required for subsequent aerodynamic analyses. The program provides for a combination of both automated and interactive mesh generation on selected flight vehicle surfaces. The automated option requires the engineer to specify only the basic parameters defining the mesh spacing and shape, while the massive numeric data defining the geometry and location of each individual panel is automatically generated by the program using the basic surface

definitions extracted from the geometry data base. The interactive option of the module may be used by the engineer to draw the required aerodynamic mesh lines directly on a displayed view of the wing or fuselage. In either case the idealization may be viewed in 3-d (figure 3), rotated, zoomed or otherwise manipulated on the graphic display before the configuration is used in any subsequent analysis.

#### AERO - The Aerodynamic Analysis Module

The aerodynamic analysis module is based upon discrete element analysis procedures as outlined in reference 7. It solves many wing-body and wing-body-tail flow configurations over a wide range of subsonic and supersonic Mach numbers. Bodies, nacelles, engine pods or external stores are represented by a system of line sources and doublets located along the appropriate body axis. Wings and horizontal tails are represented by a system of sources and vortices distributed on panels located in the plane of the wing or tail. Interference effects between the bodies and the wings are provided by placing additional vortex distributions on the body surfaces. Output includes the aerodynamic influence matrices for various Mach numbers as well as pressure distribution and flow-visualization parameters.

#### AEROPOST - The Aerodynamics Post-processing Module

AEROPOST is an ICG module that displays various results of the aerodynamic analysis module in graphical form on the CRT. The following features are highlighted:

- Tri-metric display of the pressure distribution on the lifting surfaces, with 3-d rotate, translate, and zoom capability
- Planform display of lifting surfaces with isobar contour plots
- Display of load distribution along the isolated body
- Display of pressure distribution along and around the interference body
- Visualization of the streamlines in the vehicle flow field by virtue of a dense display of the velocity vectors at many points in any selected plane.

#### ISLADE - The Interactive Structural Layout and Design Module

ISLADE is an ICG module which greatly enhances the structural engineer's capability in designing and laying out the primary structure of a lifting surface or fuselage section of given surface geometry. The workings of this module may best be explained by resorting to an example of a delta wing:

The external geometry of the wing is first extracted from the geometry data base and its planform is displayed on the screen. The next step is to define

the principal lines of the wing, which are reference lines along which, or between which, structural design elements may be defined. The purpose of these lines is to define the main guidelines required for constructing the structural layout. The principal lines and their grid points are automatically numbered as they are defined by the designer. After defining a minimum number of principal lines (e.g. figure 4), the designer may start defining the structural elements in either of two basic methods: (a) Separate definition of each element, or (b) automatic generation of a group of elements after a minimum number of parameters describing the generation pattern are input. Thus, during the entire process, two distinct models exist: the framework of principal lines (dotted), and the structural layout consisting of the defined design elements (dashed lines). The two models may be displayed on the screen either separately or superimposed.

The types of structural design elements presently available in the ISLADE module are: I-beams, stringers, triangular, and quadrilateral thin panels. When the designer desires to define a design element grid point on one of the principal lines, he need only point the cursor in the vicinity of the principal line and the program will automatically place the point directly on the line. After defining the element grid points, their properties and material types may be keyed in.

The designer may at any time make major modifications to the structural arrangement by changing the position of element or principal line grid points, by deleting or adding elements, or by deleting and adding principal lines. Once the structural design is completed a data base defining all structural design elements and principal lines is created and may be saved for use by other ISSAS modules.

#### WEIGHTS - The Mass Properties Analysis Module

The WEIGHTS analysis module is a multipurpose collection of routines which optionally compute the following mass properties:

- Given an ISLADE design data base, the total weight, CG location and moment-of-inertia matrix of the structure is computed, including non-optimum allowances.
- Given a structural finite element idealization from a SMOG data base, a diagonal (lumped) mass matrix is computed in terms of all or a selected number of structural degrees of freedom. Total weight, CG location and moment-of-inertia matrix of both the idealized and lumped structure are also computed. All computations include non-optimum weight factors.

## SMOG - The Finite-Element Structural Modeling Module

SMOG is an ICG module which greatly enhances the capability of the engineer in generating a finite-element math model required for ensuing structural analyses. Grid point positions and element topology may be generated by selective or combined use of interactive and automated methods. The automated option requires the engineer to specify only those basic parameters which define the mesh boundaries, spacing, and pattern. The interactive option may be used by the engineer when no recognizable pattern is evident. In this mode he may generate grid points or elements directly on the screen with the aid of the cursor. When a fuselage-type structure is being idealized, another SMOG feature allows for automatic development of the structure into a two-dimensional surface (e.g. figure 10), upon which either of the two modeling options may again be employed. Irrespective of the modeling option used, the engineer need not at any stage concern himself with such matters as grid point numbering and coordinates.

When the math model is completed, or during any stage of its generation, the idealized structure may be visually checked and interactively corrected in a manner similar to that described in reference 8. The displayed picture may at all times be rotated in space, zoomed, translated, windowed, etc., and views of the structure with selected element types may also be displayed either separately or selectively superimposed. Examples of such displays are depicted in figures 5-8. Application of the SMOG module in the ISSAS environment can save weeks and sometimes months of valuable calendar time per design as compared with the use of manual procedures for finite-element math model preparation and checkout.

## LOADS - The Integrated Loads Analysis Module

The LOADS module generates the design loads for the flight vehicle as a superposition of aerodynamic, inertia, and other loads. These are calculated only for cases on the periphery of and within the flight envelope which are deemed to be critical. Data for these cases are obtained from the master data base in conjunction with other required input data extracted from previously formed AERO, WEIGHTS, and SMOG data bases. The LOADS module output consists of (a) Net panel loads on the lifting surfaces, optionally represented as discrete forces acting at the structural grid points, (b) Discrete reaction forces between the lifting surfaces and fuselage, (c) Load distribution on the fuselage, optionally represented as discrete forces acting at given fuselage stations, (d) Shear and bending moment diagrams for the fuselage, (e) Envelopes of the critical cases that were analyzed.



## FESA/FSD - The Finite-Element Structural Analysis and Automated Design Module

The FESA/FSD finite-element structural analysis and design module revolves around an efficient scaled-down version of an IAI batch mode analysis program. It is based upon the displacement method, and it features advanced element technology in conjunction with efficient solution algorithms. By virtue of its dynamic storage allocation feature, small problems may be executed with minimum core-storage requirements, while large problems may be allocated larger core blocks. As is the case with all other IAI finite-element programs, FESA optionally accepts input data in NASTRAN format. This provides for full compatibility with this large scale program that is presently being employed as the major batch mode structural analysis tool at IAI. Input data for FESA is extracted directly from the SMOG and LOADS data base previously produced for the structure.

The FESA module may be employed either as a 'one-shot' analysis tool, or as the basic module in the automated resizing process of the structure. Two automated optimization modules are presently available: FSD (Fully Stressed Design) and DLD (Displacement Limited Design). The dominant resizing step is usually determined by the first fully stressed design cycle. These results may be displayed, and based upon them, the engineer may either introduce modifications or allow the FSD routine to continue, with the option of being able to display the results (using the GRASP module) after each resizing. If displacement constraints are also placed on the structure and the engineer finds that they are being violated, he may reroute the process via application of the DLD routine that will reduce the critical displacements by increasing the structure's stiffness. In this way the advantages of both automated optimization and interactive modifications are combined to efficiently produce the required structural design.

## GRASP - The Graphics Augmented Structural Post-Processing Module

GRASP is an ICG module which displays structural response parameters resulting from the FESA/FSD analysis. Selected element sets of the structure under consideration may be displayed with superimposed values of stress or force components on each element. For example, the stress components in a quadrilateral membrane may consist of one or more of the following displays: (a) Normal and shear stress components at the elements' midpoints, or optionally at the midpoints of the elements' sides, (b) Principal stresses and directions at the elements' midpoints, or optionally at the midpoints of the elements' sides, and (c) A plot of iso-stress contours for any (or all) stress component(s). Figure 9 depicts a delta-wing skin idealization with superimposed shear stresses at the center of each panel.

Figure 10 depicts a developed surface of an aircraft aft-fuselage finite-element idealization showing superimposed values of maximum and minimum



stresses in the rod elements. Figure 11 depicts a typical idealized bulkhead of the same aircraft with a plot of the stress distribution superimposed over the bar elements.

Structural deformations may be displayed with the GRASP module in two fashions: (a) Vectors proportional to the displacements at the grid points are superimposed at those points over the undeformed structure, (b) The deformed structure geometry is displayed (solid lines) over the undeformed geometry (dashed lines).

Loads data may also be extracted from the LOADS module data base, and various load distributions may be viewed superimposed upon the structural idealization. All pictures displayed in the GRASP module may be zoomed, rotated and otherwise manipulated for viewing expediency.

#### FAMOS - The Frequencies and Mode Shapes Module

The FAMOS module computes the natural frequencies and mode shapes of the structure employing eigensolution procedures described in reference 9. The program is based on an efficient automatic reduction scheme whereby the lower modes of structures with many degrees of freedom can be accurately extracted from a tri-diagonal eigenvalue problem whose size is of the same order of magnitude as the number of required modes. The process is effected without arbitrary lumping of masses at selected node points or selection of nodes to be retained in the analysis set. The stiffness and mass matrices required for the analysis may be extracted from the FESA and WEIGHTS data bases, respectively. The FAMOS module outputs the requested natural frequencies, the corresponding normalized mode shapes and the generalized masses and stiffness and stores them in the data base for subsequent use by other modules.

#### MODIS - The Mode Shape Display Module

MODIS is an ICG module which displays the natural vibration modes of a given structure as computed by the FAMOS module. Each requested mode shape is displayed by superimposing the zero node lines and relative amplitude vectors at each grid point, over the undeformed structure.

#### AEROLAS - The Aeroelastic Analysis Module

The aeroelastic analysis module is essentially a collection of flutter analysis programs of varying methodology and complexity. These include a supersonic flutter routine which employs the MACHBOX technique, and subsonic analyses using kernel function, doublet lattice or simple strip theory methods. The engineer selects the method to be used, and a post-processor within AEROLAS converts data extracted from WEIGHTS and FAMOS to the required input form.

## COMPUTER HARDWARE CONFIGURATION

The ISSAS system is presently supported by a hardware configuration consisting of an XDS Sigma-7 computer which drives a total of 50 teletype and 10 DVST graphics terminals supported by the CP-V time-sharing operating system. Peripheral I/O equipment consists of the standard devices, including a large high-speed drum plotter. Direct data links connecting the Sigma-7 to an automatic drafting machine and a Gerber IDS (Interactive Design System) are also being planned. The DVST (Direct View Storage Tube) display terminals are Tektronix 4010 (11") and 4014 (19") models operating from remote locations over voice-grade 1200 baud communication lines. On-line hard copy units are used with most of the terminals. The mainframe computer is to be replaced within the next year by a much larger and more powerful machine, and time-shared C.A.D. capability is expected to be enhanced.

IAI has been operating DVSTs for the past five years, and much experience has been gathered with their effective utilization. It has in fact been IAI's experience that most picture editing operations, including 'pick' and 'track' functions, may be effectively carried out with a DVST by clever use of the cursor or data tablet in conjunction with appropriate software. However, one of the known disadvantages of the DVST is that the non-refreshable nature of the CRT requires the picture to be repainted each time an item is deleted or modified. At the slow communication rate of 1200 baud, dictated by the present mainframe configuration, this has proven to be a somewhat frustrating procedure in cases where busy pictures are to be edited. Another shortcoming of the present configuration is that operations such as 3-d rotations must be software executed in the Sigma-7, with computing time being shared among the 50 users.

A study of IAI's present and future C.A.D. requirements has revealed that utilization of 4014 displays operated at their capacity rate of 9600 baud, would present a satisfactory and cost-effective solution to most of IAI's ICG needs. Higher performance refreshed CRT terminals would be required for the remainder. In the light of this study, IAI is in the process of evaluating a DVST configuration in which a Tektronix 4014 is connected to a 32K word low-cost mini-computer via a 9600 baud line. The mini in turn is connected to the Sigma-7 at 1200 bauds. This configuration allows the user at the DVST terminal to transfer a picture from the Sigma to the mini-computer, and then to carry out picture editing operations using the compute power of the local mini, which repaints the pictures at 9600 baud.

IAI has also been evaluating an Imlac PDS-4 graphics display system. This is a refreshed CRT display with a mini-processor, with interaction by means of a keyboard and light-pen. The remote PDS-4 is linked on-line (at 1200 bauds) in a time-sharing mode to the Sigma-7. An engineer sitting at the display may communicate with the Sigma in a time-sharing mode as if it were a teletype. He may at any time transfer a picture file to the PDS-4 memory and locally edit or otherwise manipulate it (figure 12), while benefiting from all the capabilities of a refreshed CRT with a local processor. The PDS-4 is being considered as a replacement for the DVST display as the graphics terminal for the ISSAS system.

## FUTURE DEVELOPMENTS

Two of the main principles in the development philosophy of the ISSAS system have been its modular construction and its computer independence. Preliminary studies are already under way at IAI with the objective of modifying certain modules or completely replacing them by others featuring more advanced technology. The modularity and flexibility of the system make this a straightforward and routine task. Thus, ISSAS is presently considered as a pilot system, which over the years is expected to grow, mature, and continuously extend its capabilities according to the advancing state of technology.

Further stages in the development of a major optimization capability within the ISSAS system will involve work along two major paths: (1) Derivation and application of additional optimality criteria for other response phenomena such as buckling, divergence, and flutter, and (2) The development of optimization programs based on mathematical programming methods to deal with more sophisticated classes of variables, e.g. geometry and topology. A rational blend of these two types of optimization techniques with interactive capability will provide a powerful tool for the design optimization of aircraft structures in future levels of ISSAS.

As part of the general program of introducing C.A.D. methods and technology at all levels of the design/analysis/manufacturing process at IAI, development is also presently being undertaken in the area of CADD (Computer Aided Design Drafting). In a predictable effort to increase the interactive design capability in ISSAS, the future will most probably see a 'coming together' of both systems via the computer. Thus, while certain analytic capabilities in ISSAS may be employed in the detail design of structures, certain interactive design routines and data bases are sure to be 'lifted' by ISSAS users from the CADD system.

As more and more modules are added to ISSAS, and with the growing size of the user community and number of new projects, it is expected that major efforts in the future will also be placed on the development of an advanced executive control system for data base management. This effort is expected to start within the coming year with the replacement of the present computer configuration.

Another foreseen development which will aid the ISSAS system is the implementation of an advanced high-level Fortran-based graphics language, which would appear to be device-independent to the user. One such language is presently under evaluation at IAI.

Finally, an additional area of increased capability for ISSAS is provision for the use of flexible aircraft loads in the design and analysis of primary aircraft structures.

## CONCLUDING REMARKS

An overview of an interdisciplinary structural design and analysis system which is being developed at IAI has been presented. As emphasized throughout the presentation, the use of graphic display terminals is the key to the success of such a system, since it provides the most natural and most effective mode of communication between the engineer and the computer. The engineer is then free to concentrate on his primary role as key decision maker, being able to observe the progress of the design and analysis process as it is presented pictorially on the screen, while having the option of altering its direction when undesirable trends are observed.

The ISSAS system is presently in advanced stages of development. Many of the modules which have already been completed are presently in operational use at IAI and their merits have been established. Other modules are in various stages of development and have yet to be proven in practical use. The entire system operating as an integral unit has yet to be fully evaluated, and any final conclusions regarding expected savings in engineering and computer costs would at this time be premature.

## REFERENCES

1. Wennagel, G. J., Mason, P. W., and Rosenbaum, J. D.: " IDEAS, Integrated Design and Analysis System", Society of Automotive Engineering, Preprint 680728, October 1968.
2. Giles, G. L.: " A Procedure for Automating Aircraft Wing Structural Design", Journal Struct. Div., ASCE, January 1971, pp. 99-113.
3. Giles, G. L., Blackburn, C. L., and Dixon, S. C.: " Automated Procedures for Sizing Aerospace Vehicle Structures (SAVES) ", Journal of Aircraft, Vol. 9, No. 12, December 1972, pp. 812-819.
4. Sobieszczanski, J., and Loendorf, D.: " A Mixed Optimization Method for Automated Design of Fuselage Structures", Journal of Aircraft, Vol. 9, No. 12, December 1972, pp. 805-811.
5. Backman, B. F., et al: " Aircraft Strength and Stiffness Design Automation", USA-Japan Design Automation Symposium, August 1975, Tokyo, Japan.
6. Batdorf, W. J., Holliday, J. F., and Peed, J. L.; " A Graphics Program for Aircraft Design - GPAD System", AIAA Paper 75-136, January 1975.
7. Woodward, F. A.: " Analysis and Design of Wing-Body Combinations at Subsonic and Supersonic Speeds", Journal of Aircraft, Vol. 5, No. 6, Nov.-Dec. 1968.
8. Shomrat, J., Schweid, E., and Newman, M.: " The SAGE System", Structural Mechanics Computer Programs, University Press of Virginia, 1974.
9. Newman, Malcolm, and Pipano, Aaron: "Fast Modal Extraction in NASTRAN via the FEER Computer Program", NASTRAN: Users' Experiences, NASA TM X-2893, 1973, pp. 485-506.

[illegible]

245



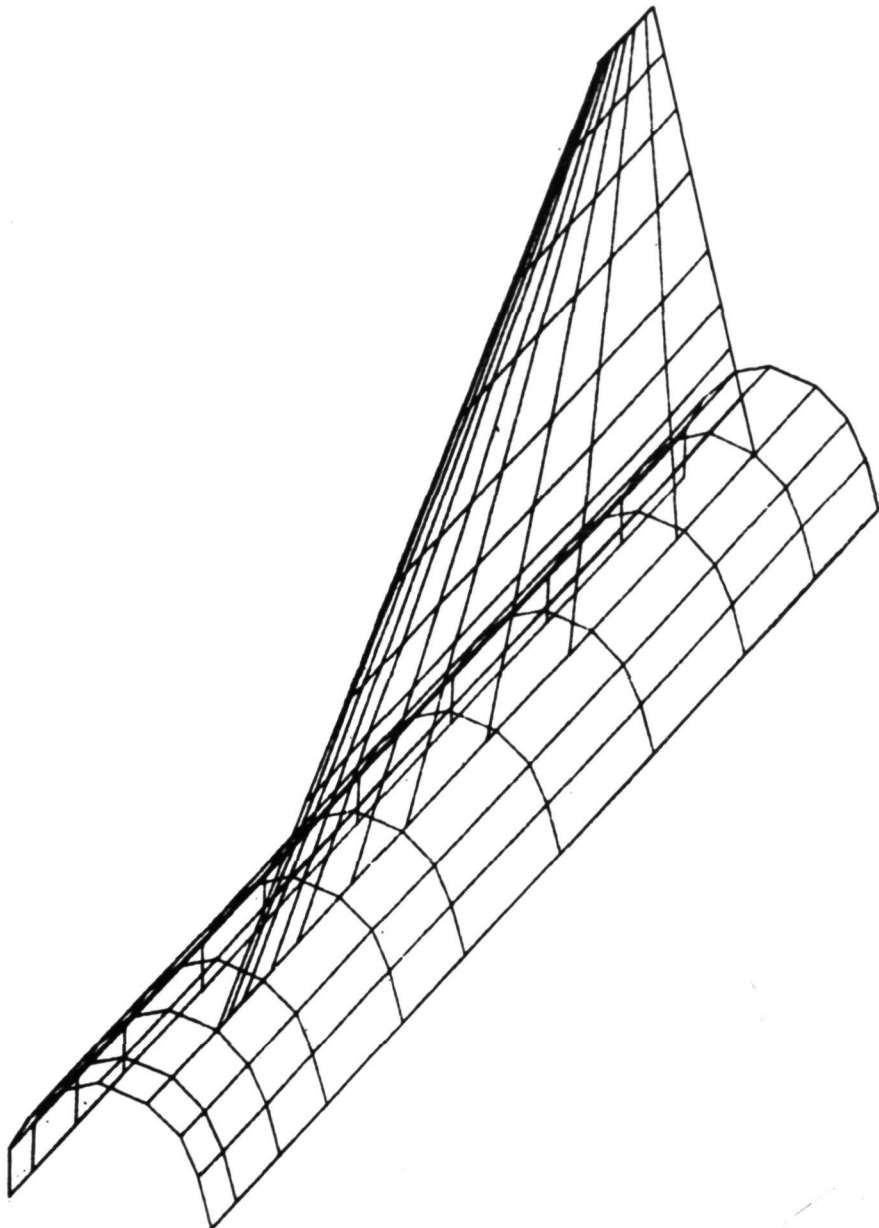


Figure 3.- Aerodynamics Modeling Module: wing-body combination.



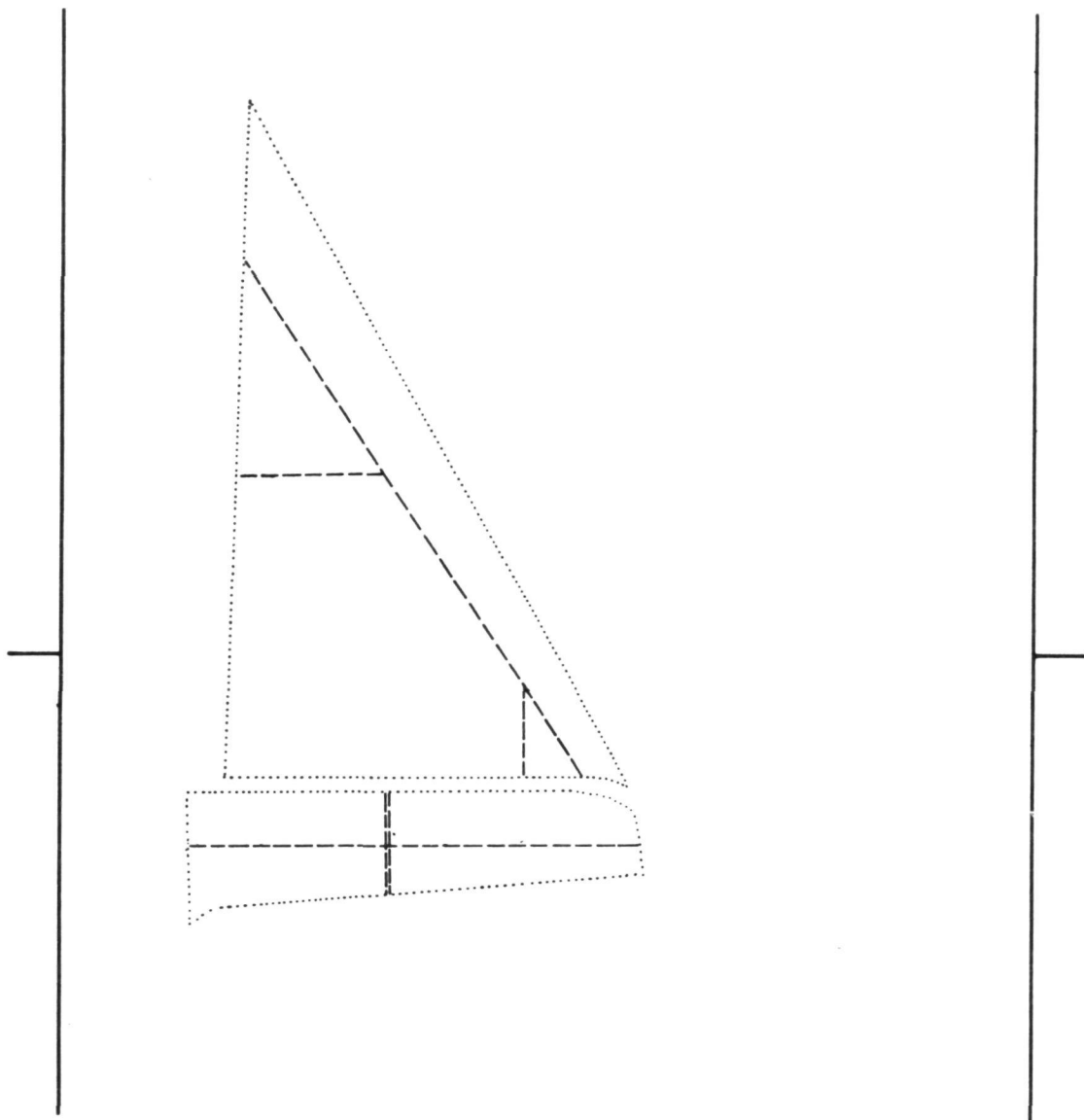


Figure 4.- Structural Design and Layout Module: delta wing.

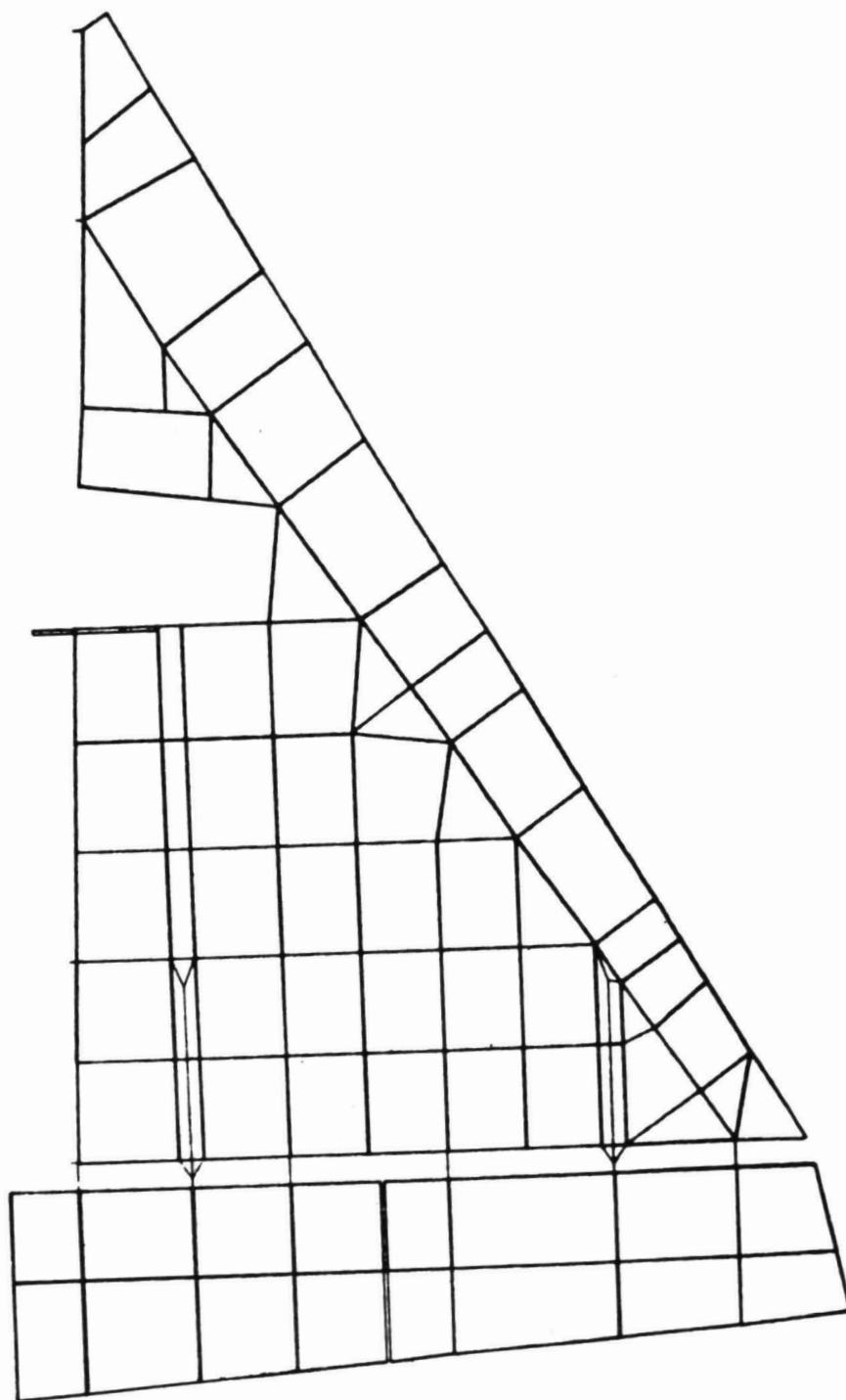


Figure 5.- Structural Modeling Module: plan view of idealized delta wing.

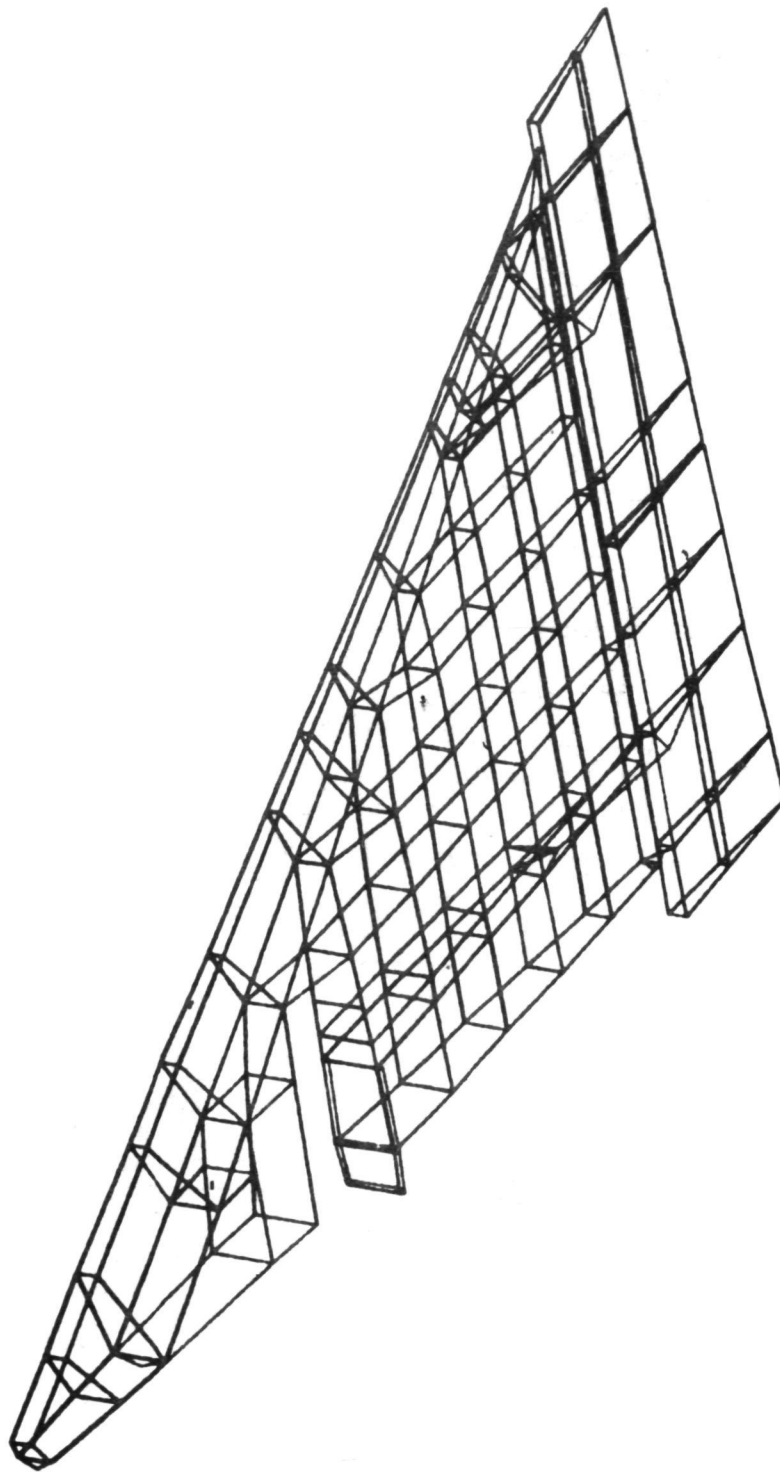


Figure 6.- Structural Modeling Module: 3-d view of idealized delta wing.

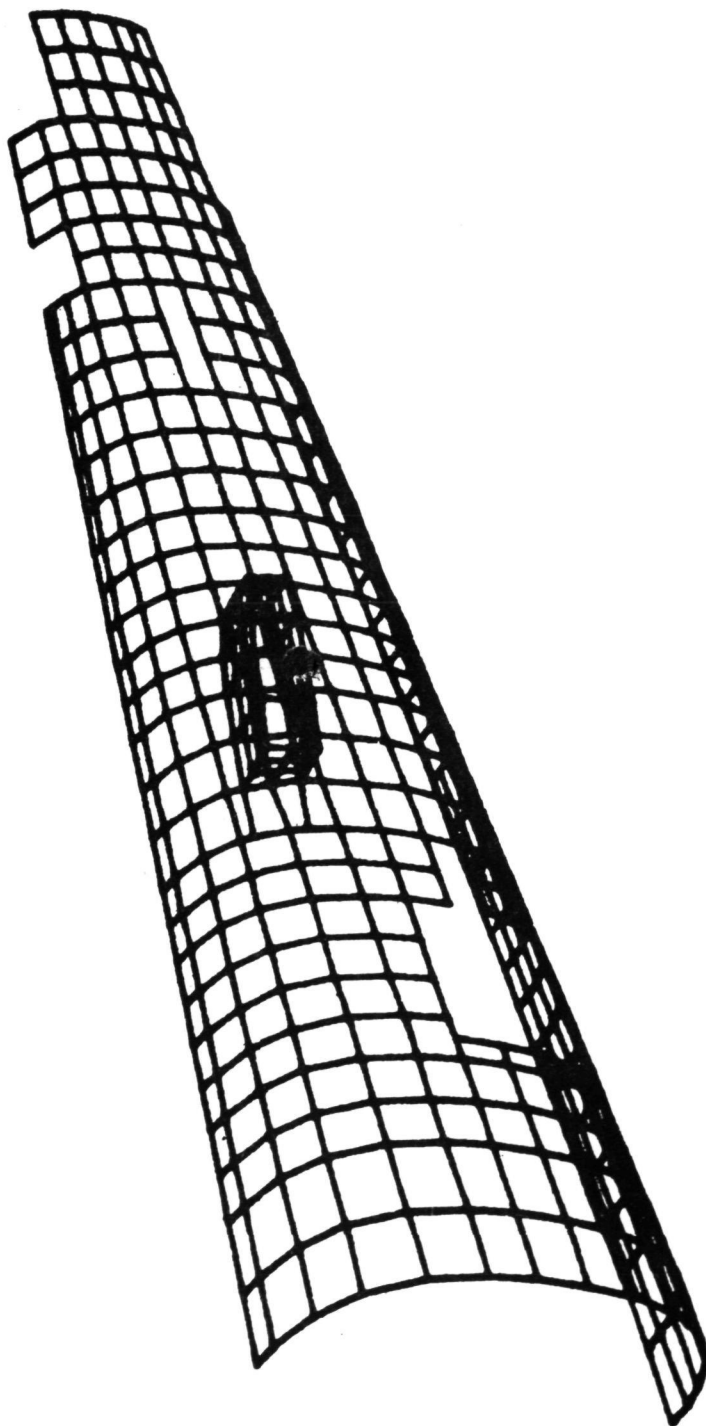


Figure 7.- Structural Modeling Module: 3-d view of westwind aft-fuselage.

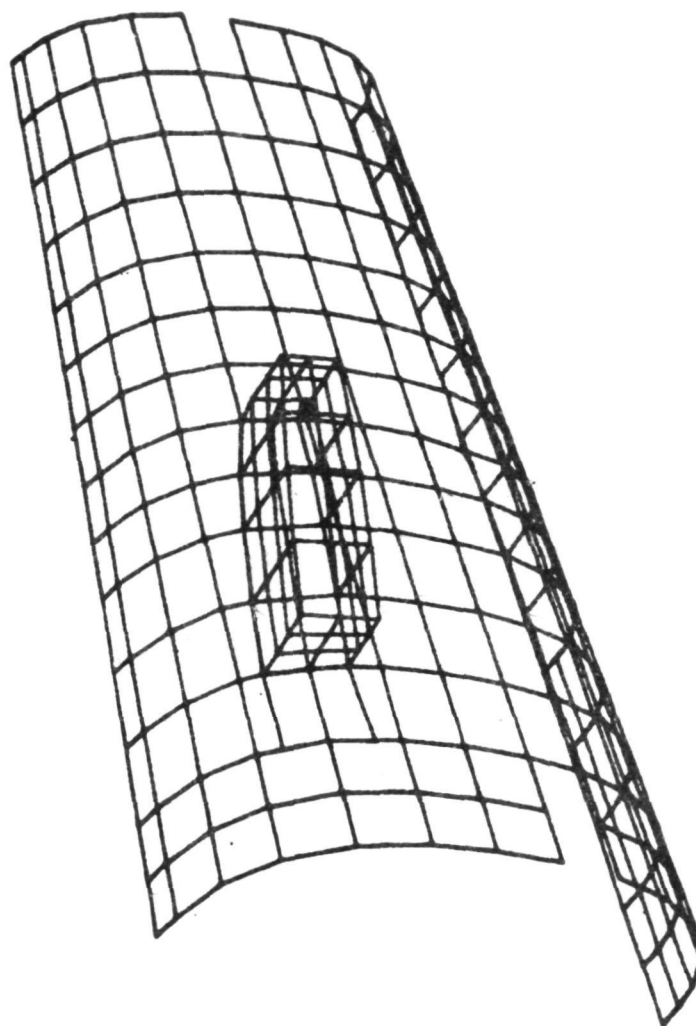


Figure 8.- Structural Modeling Module: 3-d blowup of westwind aft-fuselage.

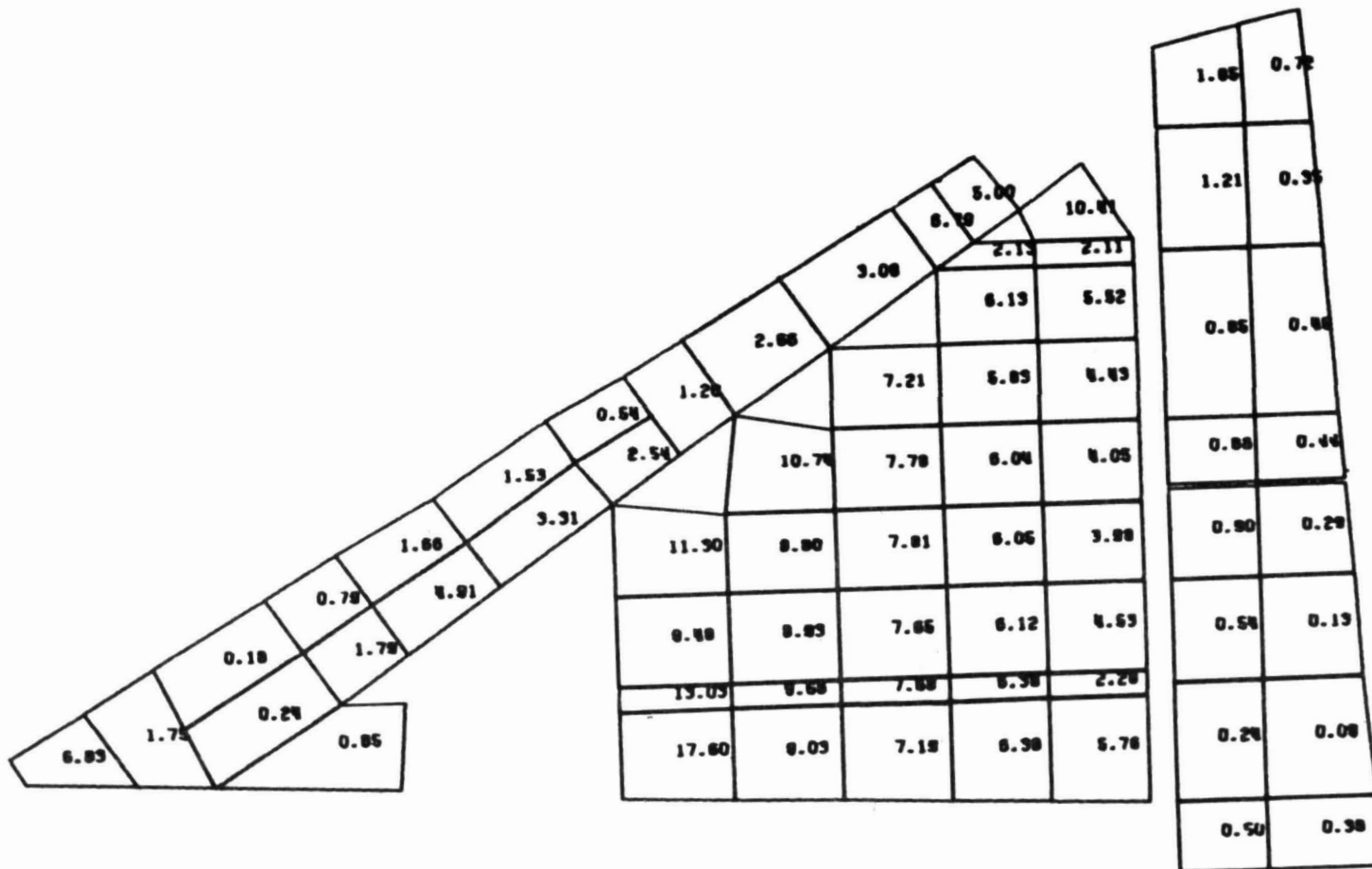
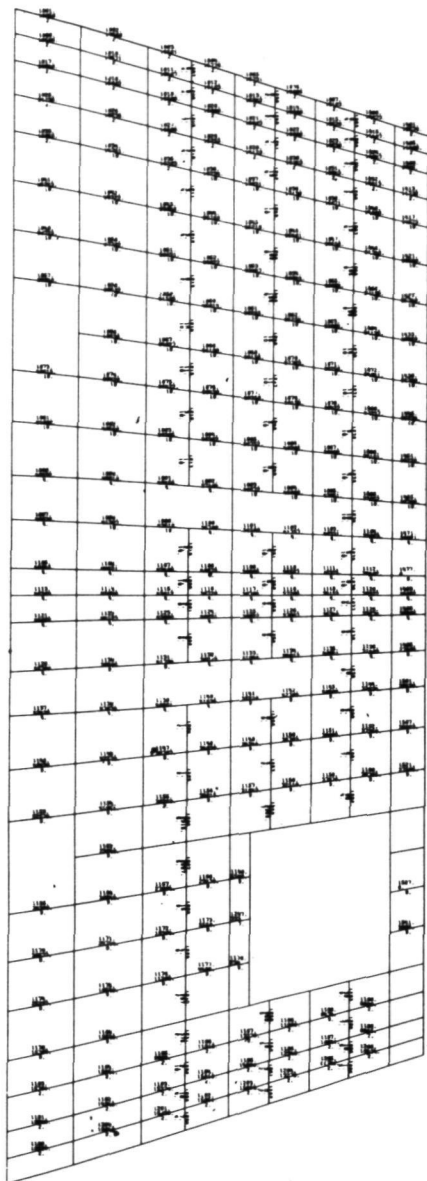
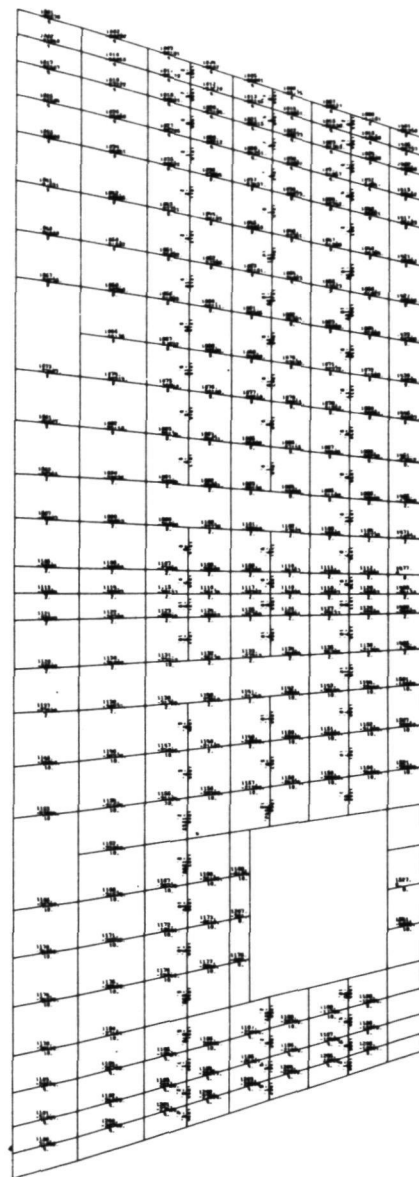


Figure 9.- Structural Post-processing Module: shear stress distribution in delta wing.



WESTWIND SUBSTRUCTURE  
ENVELOPE FOR MAXIMUM  
STRESSES IN ROD ELEMENTS



WESTWIND SUBSTRUCTURE  
ENVELOPE FOR MINIMUM  
STRESSES IN ROD ELEMENTS

Figure 10.- Structural Post-processing Module: shear stress distribution in developed surface of westwind aft-fuselage.

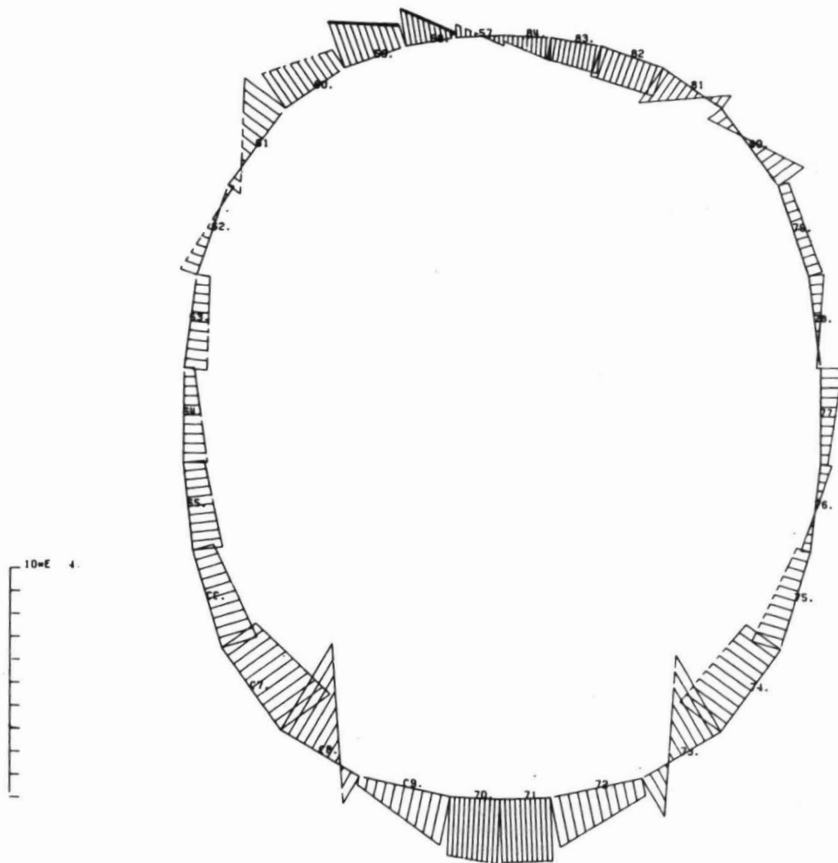


Figure 11.- Structural Post-processing Module: bending moment distribution in westwind fuselage frame.



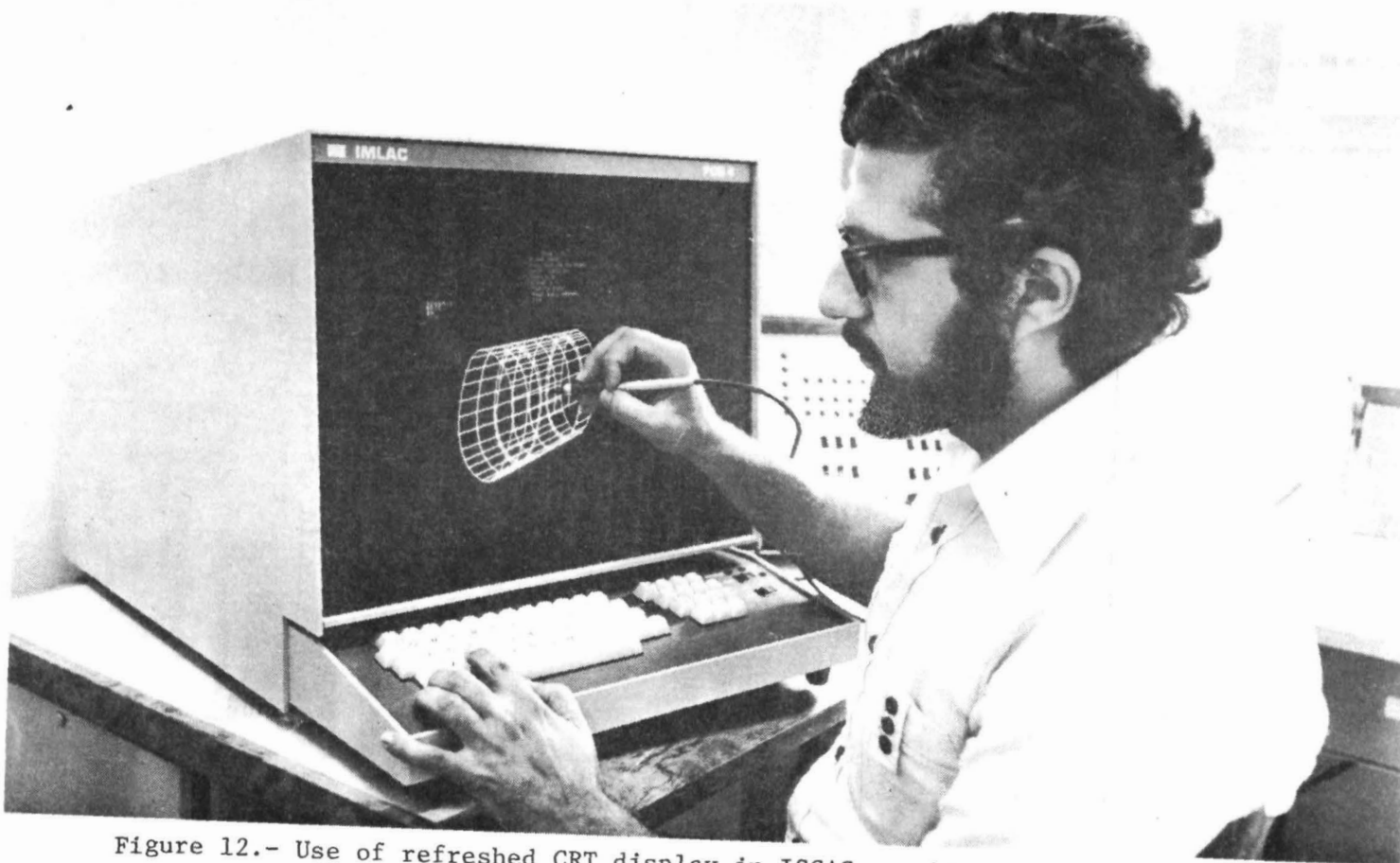


Figure 12.- Use of refreshed CRT display in ISSAS graphics evaluation program.

COMPUTER GRAPHICS APPLICATION IN THE  
ENGINEERING DESIGN INTEGRATION SYSTEM

C. R. Glatt, R. W. Abel,\* G. N. Hirsch, G. E. Alford,  
W. N. Colquitt and W. A. Stewart

Sigma Corporation

INTRODUCTION

13

The Engineering Design Integration (EDIN) System (reference 1) is a collection of computer aided design software and hardware that is being developed by the National Aeronautics and Space Administration, Johnson Space Center (JSC) on the Univac 1110 computer. A three-level system development schedule is being pursued at JSC. The Level I System, currently operational, consists of a single station, single user demand system with considerable passive graphics capability, both online and off-line. The system provides a conceptual design capability useful for advanced concept studies and limited preliminary design applications.

The Level II System now under development at JSC emphasizes interactive man-in-the-loop capabilities supported by computational interactive hardware remote to the host computer. The Level II System is also expected to provide greater depth of analysis through improvement in the technology modules (i.e. structural sizing based on flight loads). The current development will provide the prototype for the Level III System, a multi-station multi-user system, expected to support many projects and many design activities. To achieve the goals set for the EDIN System, considerable expansion of the technological capabilities, the interactive functions, the executive capabilities and utility programs is planned.

This paper deals with one aspect of the EDIN development, computer graphics and its application to design problems. Several conceptual design studies have been supported at JSC using the EDIN System and a variety of computer graphics technologies have been employed in the ensuing analysis.

Three basic types of computer graphics have evolved in the development of the EDIN System; offline graphics systems using vellum-inking or photographic processes, online graphics systems characterized by direct coupled low cost storage tube terminals with limited interactive capabilities, and a mini-computer based refresh terminal offering highly interactive capabilities. Each of these basic types of graphics has been employed to some extent. The results of using the graphics capability will be discussed as it relates to the application and continued development of the EDIN System.

---

\*Mr. Robert Abel of NASA/Johnson Space Center, Engineering Analysis Division was the Contract Monitor for EDIN development contracts NAS9-12829, NAS9-13584 and NAS9-14520.

## THE EDIN SYSTEM

The EDIN System is a computer aided design complex shown schematically in figure 1 for the evaluation of aerospace vehicle preliminary designs. The system consists of the Univac 1110 computer employing the Exec 8 operating system, a set of demand access terminals of both the alphanumeric and static graphics types, a minicomputer based interactive graphical display system and a library of independent computer programs. The program library contains technology oriented programs for estimating all major flight vehicle characteristics such as aerodynamics, propulsion, mass properties, trajectory and mission analysis, cost, steady state aeroelasticity, flutter and stability and control. There are special and general purpose utility programs in the library for generating, analyzing and controlling the flow of design data within the computer program complex.

The independent program approach to integrated design allows any of the library programs to be revised, extended or replaced without affecting the other program elements of the system in any way. External to the technology programs themselves, a data base is maintained by a series of data processors exemplified by reference 2. Each technology program may draw upon the data base for information as required. The specialists in any technology area are able to phrase the analysis of the design without regard for the other technologies involved other than the interfaces with the data base. The data base attributes are defined by the design staff and can consist of all information which is communicated between elements of the system or communicated from the system itself to the staff. The stored data represents a subset of the total amount of information generated by all of the programs within a given design sequence. When combined with the normal input data, it is sufficient to completely define the vehicles under study. When combined with the normal output, it represents the data description of the vehicle, its characteristics and mission requirements.

The total data base consists of program elements, data elements and design procedural elements stored on the Univac mass storage system. The elements are constructed in name addressable regions of online storage which can be further subdivided into user established classes of data. Each class represents an independent subset of the data recallable by the user of the system. Data manipulation is performed by several processors, some supplied with the Exec 8 System and some developed specifically for use in the construction and maintenance of the data base by the user.

The EDIN System can be operated in a batch, demand or interactive environment (figure 2). In the batch mode all logical decisions must be built. In the demand mode some user interaction with the data base is available but little interaction within a technology module execution is provided. In the interactive mode, the user has the ability to interrupt the analysis process, perform alternative analysis and alter the course of the design process. Usually a combination of batch, demand and interactive modes of operation is employed in the use of the EDIN System.

## THE COMPUTER

The host computer selected for EDIN is the Univac 1110 computer consisting of the executive system (Exec 8), compilers, system utilities, subroutine libraries, etc. The EDIN System relies heavily on the executive functions for program control and file management. For this reason, a knowledge of the control functions of Exec 8 are essential to the user of the EDIN System.

The operating system is an outgrowth of Univac's many years of experience in multiprogramming, multiprocessing, time sharing and communication systems. It provides a flexible user environment which is essential to the operation of the EDIN System. A complete set of software, ranging from high level language compilers to basic service functions, is included in the operating system. The five major categories are:

- System Processors.
- Language Processors.
- Utility Processors.
- Subroutine Library.
- Applications Programs.

The first two categories represent the base complement of software supplied with the 1100 series computer and maintained at the Univac installation. They are generally used by the EDIN System without modification. The user portion of the operating system is described in the last three categories. The EDIN System takes advantage of the software which is already available and augments the software capability in these categories with special programs dealing primarily with engineering and design integration. The EDIN interface to the Univac 1100 series computer is shown in figure 3.

### The Executive System

To take full advantage of the speed and hardware capabilities of the 1100 series system, a comprehensive internal operating environment is provided in the Exec 8. This environment permits the concurrent operation of many programs and directs the computer to react immediately to the inquiries, requests and demands of many different users at local and remote stations. The Exec 8 system can store, retrieve and protect large blocks of data and makes the best use of available file space.

Only through central control of all activities of the system can this environment of combined hardware and software systems be fully established and maintained to satisfy the EDIN requirements. The responsibility for centralized control is borne by the Exec 8 system, which controls and coordinates the functions of the internal environment. A relatively simple interface to the executive, based on the run stream concept, is provided which relieves the users of concern for the internal interaction between his program and other coexistent programs.

The technical capabilities of the Executive System cover a variety of data processing activities. The executive system design offers the versatility of handling batch processing, demand processing and real time processing using multiprogramming and multiprocessing techniques. Batch jobs may be submitted from remote terminals as well as from central site equipment. The batch mode is used in EDIN applications for long running analysis and programs with high core requirements.

Complementing the batch processing capabilities are the Exec 8 demand (or time sharing) capabilities. This mode of operation accommodates inquiry terminals operating in a demand mode. All facilities available to the batch processing user are also available in a demand mode, the primary difference being that the demand mode permits the user additional flexibility in the statement and control of individual tasks. For example, when an error is made in the demand mode, the user simply corrects it online and proceeds rather than suffering the turnaround cycle inherent in batch processing. The demand user may interact with the system at various levels which include the Executive System, a conversational processor or with a user program. Many conversational EDIN programs take full advantage of the demand mode. Design analysis can be interrupted at any point to assure integrity using the demand mode.

The Exec 8 system is also designed to be used with other computers which have parallel processing and intermediate data transmission requirements. The Univac Communications Subsystems, together with the scheduling and interrupt processing features, provide an environment satisfactory for the operation of parallel processing computers. The EDIN System currently uses this capability for the interactive graphics (Adage 330) computer.

The Executive System is designed to ensure effective and efficient utilization of the mass storage devices. The consequence is an unprecedented ability to relieve users of the responsibility of maintaining cards and magnetic tape, thus eliminating many errors which heretofore have accompanied the use of large scale software systems. At the same time, the overall operating efficiency is considerably improved. Card handling is virtually eliminated in the use of the EDIN System.

#### Executive Control Statements

Control of the operating system for the Univac 1100 series computer is accomplished through a relatively small and easily learned set of executive control statements. These statements which are identified by a leading @,

@statement

direct the executive system in the processing of a run. Control statements may evoke executive functions such as scheduling, assignment of facilities (files), etc. or may cause the execution of a user program or a processor. The executive control statements are designed in a

compact and descriptive manner to facilitate use and yet provide access to all of the features of the Executive System, the EDIN program library and Data Management System.

### Run Stream Concept

The run stream concept is employed as the primary interface between the computer operating system and the user. The run stream contains control statements and data that describe the specification of the tasks which will be performed by the computer. It is the largest working group read and manipulated by the Executive System. The run stream itself is a sequence of data images which taken as a whole constitute the total specifications for a run. The run stream consists of an @RUN control statement followed by other control statements and data which direct the performance of individual tasks. Each task consists of one or more control statements which provide technology oriented or utility operations. Tasks may also be grouped into partial run streams. The partial run stream may be added at any point in the run stream by use of the @ADD control statement. The run stream is terminated by an @FIN card which directs the executive to terminate processing of the run stream. In the batch stream of operation, the entire run stream is normally stored on the mass storage facilities before run processing is initiated. The Executive System schedules the run stream as an unit. Once initiated, the executive processes the entire run stream. In a demand mode, the run is normally initiated immediately upon acceptance of the @RUN control statement from the demand terminal. The system continually solicits the demand terminal for additional run stream input which generally occurs dynamically or on a conversational basis. The demand solicitation continues until an @FIN control statement is submitted from the demand terminal.

Once the run is opened, the executive processes the control statements as they are encountered. A batch run terminates as the result of an abnormal task termination. However, in the demand mode, an abnormal task termination will not terminate the run but simply print the diagnostic message which causes the abnormal task termination and solicits another executive control statement.

### EDIN/EXEC 8 INTERFACE

The EDIN System is interfaced with Exec 8 through a relatively simple application of the partial run stream (@ADD element) concept. Figure 4 illustrates a typical conceptual sizing study involving a multiple stage launch vehicle. Each block in the diagram represents a design task. Each task is physically constructed in an @ADD element containing control statements and data for execution of one or more programs required to perform the task. The @ADD elements are actually augmented by data base information requests. The requests are satisfied by preprocessing the @ADD element with the EDIN data processors. Once the data base requests are satisfied, the modified @ADD element may be processed by the Exec 8 system.



Data base interface is illustrated in figure 5. A partial run stream is interrogated by a data processor, which modifies the @ADD element with data base information. The modified @ADD element is then processed by the Exec 8 system resulting in generation of normal output and EDIN interface data. The interface data is a subset of the normal output which can be placed in the data base by a data processor.

Any set of vehicle component matching and sizing loops can be defined by one or more partial run streams (or design tasks). There is no effective limit to the number of design sequences which can be performed using the partial run stream concept. The sequences can be repeated as often as necessary and can be terminated on criteria contained in the data base.

#### DATA MANAGEMENT SYSTEM

The EDIN System provides a balance of data management techniques which consider the inherent capabilities of the computer operating system, past efforts in the storage and retrieval of stratified data and the recent development of some flexible paging techniques for the transfer of information between the computer core and the mass storage of the computer. The Univac Exec 8 System provides the resources for the storage of large complex data files, for the storage and retrieval of the files and for the cataloging protection and backup of the files. The Exec 8 System has several processors with instruction sets for manipulating the data retained in mass storage. A limitation on the operating system capabilities arises in accessing the subfile level of information in the system files once the file is addressed.

The EDIN Data Management System is designed to subdivide the Exec 8 files in a manner that will allow the data retained in mass storage to be accessed at any level from the single parameter level to a large matrix of data. Rather than constructing an extensive single computer program that attempts to be all things to all people, the EDIN Data Management System provides an extendable three-level data management capability. This approach permits the individual designer using the system to make his own decisions with regard to the storage method and techniques. It also permits the flexibility of using existing data sources not specifically created for EDIN.

The lowest level of the Data Management System deals with the interface between the data in mass storage and the computer operating system. This level of the Data Management System is provided by the Exec 8 software and consists of the file utility processors, the file administration processors and other system level processors. The system processors are accessed using Exec 8 control statements. Therefore, first level software may be used directly by the EDIN user for transmitting large structured blocks of data. The files may also be accessed by the programmer who seeks economy above all else. The file level constitutes the foundation for all higher level data management components.

The second level of the Data Management System provides the mechanism whereby the files can be organized into blocks of data called pages. Pages of information can be organized in a number of ways and names can be given to each page. A pointer system or directory is maintained by a Fortran callable software package developed for and maintained in the EDIN library. The second level is the programmer level. It is used to interface the technology modules directly with the data base and constitutes the primary interface for the EDIN data processors.

The third and highest level of the Data Management System is provided to make the system more usable to the designer who may not be a programmer. The capability is provided in the data processor, described in reference 2, which is designed to maintain a data base of stratified information. The stratified data can be selectively accessed and merged with the input stream of the EDIN technology programs. This level also provides the interactive language structure which allows the user to sit at a remote terminal and interact with the data base directly and generate design reports. The DLG processor also contains routines for processing the output from the technology programs for the storage of design information in the data base.

Although the user may access the data base through any of the three levels, it is the lowest level maintained by the Exec 8 System which actually stores and retrieves the data. Exec 8 handles all of the underlying data management functions including file assignments, file directory maintenance and security procedures as well as the data block transfer to and from mass storage.

#### THE PROGRAM LIBRARY

The EDIN library of independent computer programs provides the system user with the basic building blocks to perform integrated vehicle design analysis. Thus, to establish a given analysis, it is necessary to use only those programs required for the particular analysis. Figure 6 illustrates the types of analysis that can be performed in each technology area. The programs generally perform some engineering analysis function such as aerodynamics, performance or structural analysis. For example, the aerodynamics estimation programs provide theoretical and empirical methods spanning most aerospace vehicle shapes in common use over a complete Mach number spectrum. The performance programs provide vehicle performance methods ranging from simplified segmented mission analysis to the most complex variational optimizational techniques. Although the EDIN technology module library has extensive analysis capability, it does not permit computerized design studies much beyond the conceptual level.

Other nontechnology oriented programs, called utility programs, perform computer aided analysis function such as optimization, display graphics and report writing. During a simulation, system performance and constraint criteria are generated which are later used to evaluate the vehicle design (either automatic or manual). Special purpose utility programs are aids



in this evaluation. The utility programs generally transfer information or transform data to user formats or to meet other simulation requirements. They perform such tasks as compiling and executing interface programs, suppressing unnecessary printed information generated during design simulation and the routing of design reports to alternate user sites.

Most of the technology programs were in existence before development of EDIN. The programs are largely a result of NASA or USAF sponsored research studies. Most of the utility programs were created as part of the EDIN System to facilitate integration of the technology modules. There is no limit to the number of programs which can be incorporated into the library or to the number of programs which can be called upon in a given design situation. Integration of new technology programs into the system is a relatively simple task involving only a few hours to a few days of effort. New programs are constantly being integrated into the system as a need arises.

#### THE GRAPHICS SYSTEM

The use of computer graphics has been exploited for many years in industry with specific application to geometry definition for part design. This application has been termed Computer Aided Design. Now however, computer aided design is taking on a broader meaning as more effort is being expended in the development of capabilities in integrated design analysis such as the EDIN System. Integrated design analysis becomes feasible when the software and hardware are available to analyze and interact with the design process. Computer graphics in the EDIN System refers to that portion of the software system which supports the integrated design analysis using graphics output hardware available at JSC. Three types of graphics are employed, each aimed at specific hardware classes and generally toward a specific design analysis objective.

#### Offline Graphics

The offline graphics system uses the CALCOMP pen plotter, the SD4060 film plotter or the FR80 film plotter. While these devices have historically provided the bulk of computer graphics capability, the main use for them in the EDIN System is for engineering reports. The offline systems are characterized generally by high quality and poor turnaround. The resolution is better than .254 mm. The turnaround at JSC is one to four days. Further, the user can apply little judgment in the arrangement and annotation of the plotted information. All judgment factors must be applied without benefit of seeing the plotted information.

The EDIN offline software system (figure 7) is interfaced through a series of standard graphics routines which generate lines, symbols, etc. A lower level software package provides software windowing and clipping of the plot vectors to the specifications of the user. Both software packages are designed to generate plots on all three plot hardware devices.

## Online Graphics

Two Tektronix 4012 storage tube terminals (Univac 1110 demand) have provided the bulk of the online graphical analysis capability for the EDIN system since their delivery in 1974, and for a very reasonable cost (approximately \$10 000 per station). The advantage of this low cost graphics system has been the enhanced capability afforded by the "instant" visualization of the computer results. Online decisions affecting the design can be made with little interruption of the computation process. In other words, the design analysis is performed in a much smaller span of time. The major disadvantages have been the slow line speed (300 BAUD), poor hard copy and the early limitation on vector graphics input capabilities. The slow line speed severely restricts the use of menus in program control and hence the interaction of the user with the analysis program.

Early in 1975, one Tektronix terminal was adapted to a 1600 BAUD direct line to the Univac 1110 computer. This capability had an immediate and dramatic effect on the plotting capability through a five-fold increase in plotting speed, but in general, the speed remains too slow for large amounts of vector data such as total configuration geometry and the generation of menus for program control. However, the 1600 BAUD rate is quite adequate for two-dimensional plots.

Concurrent with the upgrading of the hardware capability described above, new software was made available from Tektronix which provides input graphics capability. Operating from the keyboard, a crosshair position and a one-character command can be transmitted to the computer program in execution. The program can be precoded to perform various operations based upon the information transmitted.

The EDIN online software system (figure 8) is interfaced to the Tektronix terminals much as the offline graphics system (figure 7) is linked to the CALCOMP, SD4060, etc. There is one exception, the addition of an edit feed back. This feature allows the user to edit the plotted information. The editing includes the addition and deletion of data points and text, rescaling, windowing, etc. The data can be replotted any number of times until the user is satisfied with the content. The user can then specify that the image or plot produced on the Tektronix terminal be duplicated on any of the offline devices.

The benefits to the EDIN System afforded by the online software capability are twofold. First, the user can apply the system as a monitoring device during a design simulation. In this mode, working copy is available from the Tektronix hardcopy unit. Second, the system can be used to preview and edit plotted information before investing the resources and turnaround time to generate offline plots. As a matter of practice, the online graphics system has drastically reduced the offline plotting requirements.

After several months of operation of the EDIN System using the Tektronix terminals limited input graphics capability, it has become evident that significant improvements in the edit capability of vector data already stored can be offered but there are obvious inaccuracies in entering new vector data from the terminal. For example, the accuracy of geometric data input from the scope is generally not acceptable for engineering applications such as aerodynamics analysis or mass properties evaluations. One practical solution is the use of a data tablet, companion to the scope. These devices use a stylus (or bomb site) to detect coordinates on the tablet and can be used in conjunction with overlaid drawings to accurately enter geometric data or to edit data which resides in the data base.

The data tablet can also be used very effectively for program control menus, a capability which is not otherwise practical with the line speeds now in use. The menu can be a prepared hard copy overlay for the data tablet. Raster positions on the tablet can be coded to correspond with the menu overlay. Data tablets are available in two sizes.

The smaller size is a hand-held tablet which is ideal for menus but has limited application for geometry definition. The larger size is practical for entering geometry but does not have the portability of the small tablet. The addition of a large data tablet would raise the average cost per graphics station to about \$15 000.00. Plans for future online graphics stations for the EDIN System include the use of input data tablets.

### Interactive Graphics

A truly interactive design integration system involves an intricate relationship between the user of the system and the hardware/software system itself. To be acceptable to the design community, the interactive system must provide a natural extension of the designer's own capabilities. Therefore, any system related to interactive computer aided design must be highly user oriented. The involvement of the user during the development period is essential to the success of the system. A significant portion of past EDIN development effort has been the application of the system to real engineering design problems at NASA. It is expected that the same user community involvement will be pursued for the interactive development program.

To date, computer graphics in the EDIN System has been used in the definition and modification of three-dimensional geometry descriptions for input editing, output analysis and for report preparation using two-dimensional graphics. Highly interactive capability has been limited by the availability of suitable hardware. The recent acquisition of the Adage

330 Graphics Display System by JSC has greatly enhanced the potential for interactive computer aided design.

The Adage 330 Graphical Display System (GDS) consists of Adage DPR-4 (32K of 30 bit words) minicomputer and Adage 400 micro processor. The system has an interactive graphics terminal along with some peripheral equipment such as disk, light pen, input tablets, etc. The GDS is linked to a Univac 1110 computer via a 19,200 BAUD line. When the EDIN System is fully developed, the technology modules and the bulk of the data bases will reside on the Univac 1110. Graphical analysis functions will be performed on the GDS as well as the Tektronix. The split is not yet clearly defined but some functions will be available on both systems. The user will interact through the demand system on the 1110 and through the refresh terminal on the GDS. In the display mode, the available GDS core will be used largely for buffering display data. In a nondisplay mode, small technology modules may be executed.

For some classes of problems requiring the use of the GDS, data integrity between the minicomputer and the Univac 1110 must be maintained. Therefore, the transmission of data from the 1110 to the GDS will require special handling to provide the required data integrity. The file structure for the GDS will be compatible with the system data file formats on the Univac Exec 8 System. The user will be able to specify the accuracy degradation allowed in the transmission of data to and from the GDS mass storage system.

Two types of graphics display capabilities will be required, static and dynamic. Most static graphic functions now available in the EDIN System will be required on the GDS. In addition, new graphics functions will be required to support graphical analysis programming for the GDS. The basic static graphic function (figure 9) will be equivalent to the CALCOMP sub-routines PLOT, AXIS, LINE, GRID, SYMBOL, etc. The development activities will extend these basic graphic function requirements to include a basic set of interactive graphic functions. All GDS applications programs will use this "second level" software. Obvious applications of the GDS are three-dimensional geometric manipulations, separation studies, plume impingement studies, etc. Other potential applications include program control through light pen, selection from menus, tutorials, report generation, etc.

Interaction with large technology programs is an essential goal in the development of EDIN. Therefore, the user of the GDS will have demand access to the Univac 1110. Further, the access will be achievable during execution of any GDS program. This will allow the user to effectively interact with executing programs on the Univac 1110 while performing GDS display and editing functions. Fortran callable interrupt and data transfer utilities will be available for the development of effective interaction between the Exec 8 programs and the GDS programs.

Although some Adage supplied software is available, much software must be developed to make the GDS an effective system. Some preliminary display programs have been developed since the delivery of the system in June of 1975, but the interactive capabilities thus far have been relatively untapped.

## APPLICATION OF GRAPHICS TO DESIGN SIMULATIONS

### SRB Replacement Study

The use of a recoverable liquid rocket booster (LRB) system (figure 10) to replace the shuttle solid rocket booster (SRB) system offers the potential of reduction in cost per flight, increased payload, less environmental impact and overall reduction in energy requirements. It has been proposed that the use of the basic shuttle orbiter, a modified ET and the new LRB system could achieve the potential. The concept uses the shuttle in essentially unmodified form. An RP-1 stage using three modified F-1 engines mounted aft of the shuttle ET replaces the existing SRB's. Some modification of the ET would be required to take thrust loads longitudinally along the tank rather than laterally through the tank. The expended LRB stage is jettisoned at BECO and recovered for reuse. The mission profile, figure 11, is designed to meet the shuttle mission 3-A payload requirements with sufficient propellant to fly an AOA mission.

The study, designated EDIN04, was initiated for the purpose of assessing the proposed LRB concept. The study was performed using EDIN software and represents a joint effort between NASA and Sigma Corporation. Information available from other NASA contractors was also employed where applicable. The NASA Engineering Analysis Division and the Future Programs Division were instrumental in defining design requirements and constraints. Other NASA divisions were involved in the analysis. Sigma Corporation developed the simulation procedures and assisted with the engineering analysis which resulted in a technical assessment of the concept.

### Technical Assessment Techniques

Historical weight estimating relationships were developed for the LRB using Saturn technology and modified as required to support the EDIN04 study. Based on NASA funded studies, no weight penalty for the ET redesign was assessed in the present study. Mission performance was computed using current shuttle computation ground rules to allow for reasonable comparison of the existing shuttle with the EDIN04 results. Performance analysis was based on a point design trajectory model which optimizes initial tilt rate and post BECO control history. A gravity turn was employed during the LRB boost phase rather than using the shuttle angle of attack profile. A combination of SSME throttling and one F-1 engine shutdown was used to control the dynamic pressure and the longitudinal acceleration limits.

The study was conducted in four parts. The first and basic EDIN04 assessment provides a comparison of vehicle designs with and without the RTLS/AOA trajectory constraint and one with a 10% dry weight contingency imposed on

the LRB. The second part of the study considered the possibility of a degradation in SSME thrust level. The LRB was redesigned for four levels of SSME thrust. The third part considered the possibility of off-loading ET propellant using a base line 10-meter diameter booster and SSME thrust levels degraded to 85% of currently quoted levels. The fourth and final portion of the study upgraded the weight estimating relationships resulting from modification of the interstage and aft skirt lengths, deletion of the intertank structure, modification of the LOX/RP propellant tank weight sizing coefficients, modification of aerodynamic surface sizing equations and modification of the interstage sizing coefficient. Selected study configurations from the earlier parts of the study were resized. Results of the study have been retained by NASA for internal use.

#### Computer Graphics Utilization

Computer graphics was employed in the EDIN04 study in two ways:

Two-Dimensional Analysis Plots.

Three-Dimensional Configuration Studies.

The two-dimensional plots illustrated in figure 12 were generated using the online Tektronix terminal for visual control and monitoring of the flight constraints such as load factors and dynamic pressure. The plots were also generated on CALCOMP for incorporation into the assessment report. A general purpose independent plot program of reference 3 has the ability to extract data from the data base and present it in user specified formats.

Plotting instructions are specified in the run stream or they can be specified conversationally by the user. The data may be presented in the usual x-y sequence or can be presented as contour plots. The user can select the segments of data to be plotted as well as scaling, annotation, titling and smoothing of the resultant plot. Options are available for grid lines, multiple access and for tabulating the data in columnar format. The data itself may be edited by repositioning data points, the addition of text, etc.

Three-dimensional configuration images were used for calculation of LRB aerodynamics, for launch and separation sequences and for the generation of mass properties data. The three-dimensional imaging program (reference 4) uses a surface definition based upon quadrilateral elements to describe a picture-like drawing of an arbitrary shape. The program can be operated in a demand or batch mode and can construct both online and offline images. The image geometry is used in the weight and balance program where each quadrilateral represents an element of mass which is summed with the other elemental masses to form the total mass properties of the geometric shape. Weight and balance calculations were emphasized in the EDIN04 study.

## ADAGE GRAPHICS APPLICATIONS

Since the installation of the Adage 330 in June of 1975, there have been relatively few actual applications developed. Those which have been developed have been experimental in nature providing users with software design data for future development activities. Among those applications which have been developed are the geometry input module, an editing module and a separation module. The geometry input module is illustrated in figure 13. The sequence of four photographs preceding clockwise from the upper left shows an engineer using the data tablet for two-dimensional geometry input, the resulting section input displayed on the screen, the fuselage component, which is the result of a number of section inputs and the final configuration consisting of a number of components.

The edit module is illustrated in figure 14. Here, the picture sequence illustrates a three-view (developed by Nelson Logan of Lockheed Electronics Corporation) of the total configuration with an oblique view inset which can be rotated to any angle. The other three picture sequences illustrate plan form, inboard profile and section cuts using the hardware windowing techniques available in the Adage 330 micro-processor. The window techniques illustrated in figure 14 have been standardized into a Fortran callable subroutine package for use in any Adage 330 computer program.

The separation module is illustrated in figure 15. The geometry for a number of vehicles is stored in the EDIN data base. These geometric configurations can be loaded into the separation module as individual components. The module then uses separation data generated external to the separation module. Dynamic separation sequences can be viewed from any observer orientation. The sequences shown in figure 15 illustrate SRB and external tank separations (top) for the shuttle launch sequence and the shuttle separation (bottom) from the Boeing 747. The sequences shown are illustrative in nature and do not represent actual engineering studies.

All of the above modules are preliminary software developments and will undergo major refinements during the next several months. They are discussed here to provide illustrations of the type of EDIN related analysis which can be performed on the Adage 330 computer. Other applications are being explored which take advantage of the Adage capabilities.

## CONCLUDING REMARKS

The EDIN System at JSC has provided a computer aided design environment in which conceptual and preliminary designs can be assessed in a timely manner (usually 3-6 weeks). The components of the system are the Univac 1110 computer, a flexible data management system and a library of independent technology programs. The EDIN interface to the Univac computer is formulated on the run stream concept in the language of the computer operating system and the technology module input data. Both elements of the run stream can be augmented with data base information to satisfy the interdisciplinary design requirements.



The EDIN System provides the user with the ability to formulate the computer aided design problems at the task level in the same manner as is employed in the industrial design process. However, most communication is handled by the computer. Demand interaction with the programs can be performed. Any vehicle matching and sizing loops can be constructed. A flexible data management system provides the user with the ability to construct data base elements to meet both small and large design objectives. Once constructed, the data base elements can be assessed at the user level or at the program level and the data can be secured as read only elements using the existing Exec 8 software. The resulting design simulation is a flow of program executions which are completely controlled by a man-in-the loop.

The program library contains technology modules which permit the user to phrase most design problems at the conceptual and preliminary design levels. A large repertoire of EDIN utility routines have been created for interfacing the technology programs to provide a smooth flow of data from one to the other.

A graphics system for EDIN has been created which permits offline graphics, online graphics and interactive graphics. The offline graphics provides report quality graphical information. The online graphics provides quick-look analysis information, which drastically reduces the design cycle time. The interactive graphics capability, now in the embryo development stage, is expected to speed up the initial phases of design simulation and provide more flexibility to changes in direction. Computer graphics has brought to the EDIN System a more effective analysis capability, quicker design cycle time, rapid response to user inquiry and the potential of man-in-the-loop interactive design capability.

#### REFERENCES

1. Glatt, C. R., Hirsch, G. N., Alford, G. E., Colquitt, W. N. and Reiners, S. J.: The Engineering Design Integration (EDIN) System. National Aeronautics and Space Administration Contractor Report. NASA CR-141598. December 1974.
2. Glatt, C. R. and Colquitt, W. N.: The DLG Processor - A Data Management Executive for the Engineering Design Integration (EDIN) System. Volume I - Engineering Description and Utilization Manual. National Aeronautics and Space Administration Contractor Report. NASA CR-141596. December 1974.
3. Glatt, C. R. and Hirsch, G. N.: PLOTTR: An Independent Computer Program for the Generation of Graphical Displays. National Aeronautics and Space Administration Contractor Report. NASA CR-141593. November 1974.
4. Glatt, C. R.: IMAGE: A Computer Code for Generating Picture-Like Images of Aerospace Vehicles. National Aeronautics and Space Administration Contractor Report. NASA CR-2430. September 1974.



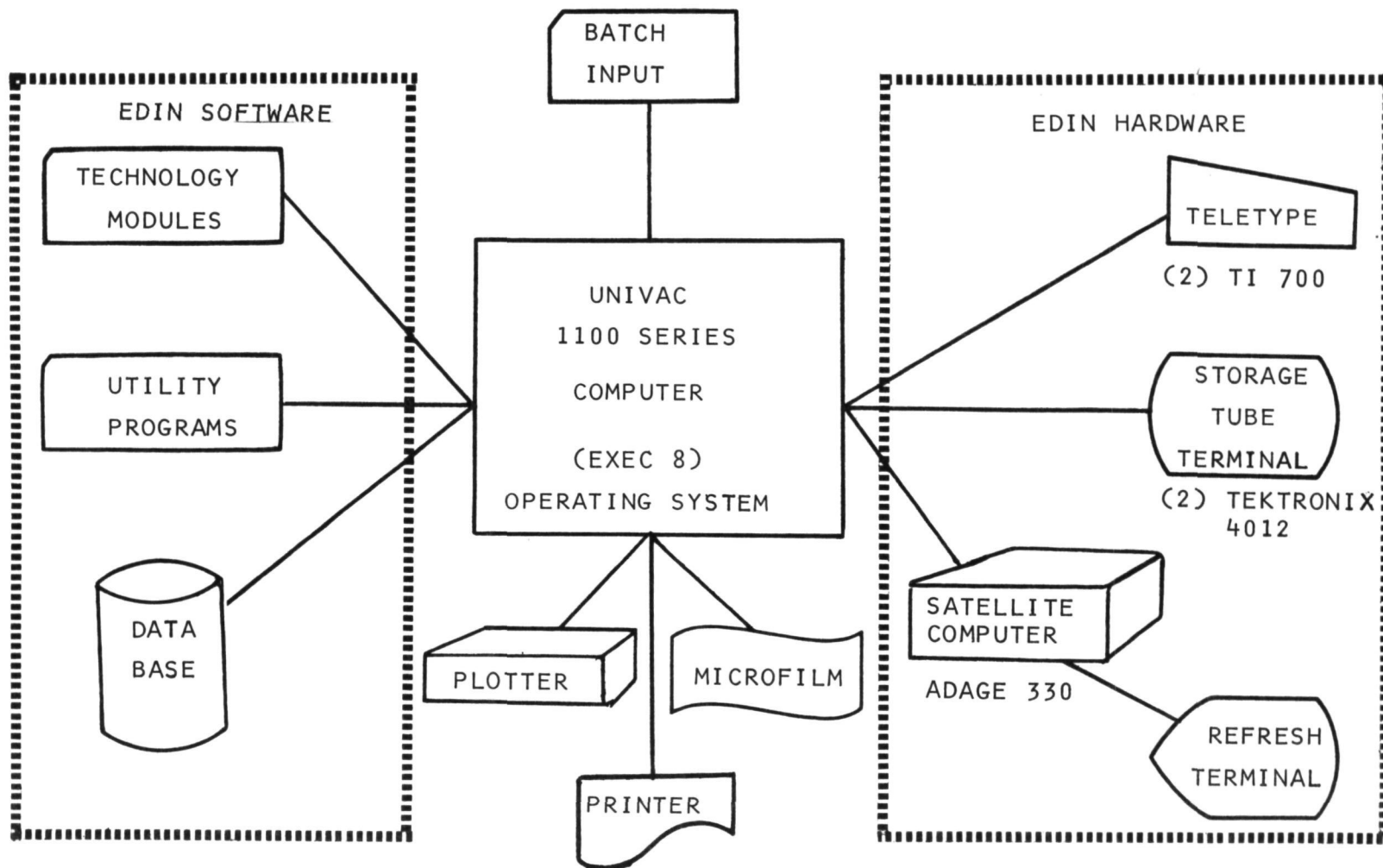


FIGURE 1

COMPONENTS OF THE EDIN SYSTEM.

## OPERATING MODES

### BATCH OPERATION

PREPARED SIMULATION STREAM  
NO INTERRUPT  
ALL CONTROL LOGIC BUILTIN  
ONSITE OR REMOTE SUBMITTAL  
OFFLINE GRAPHICAL ANALYSIS

### DEMAND OPERATION

PREPARED SIMULATION STREAM  
INTERRUPT AND RESTART CAPABILITY  
DATA BASE INTERROGATION AND EDIT  
AUXILIARY CALCULATIONS CAN BE PERFORMED  
ONLINE GRAPHICAL ANALYSIS AND EDITING

### INTERACTIVE OPERATION

PREPARED OR UNPREPARED SIMULATION STREAM  
INTERIM ANALYSIS OF RESULTS ONLINE  
ALTERATION OF SIMULATION STREAM ONLINE  
EXECUTION OF PARTIAL SIMULATION STREAMS TO SUPPORT ANALYSIS  
REPORT PREPARATION AND MODIFICATION  
DATA BASE INTERROGATION AND EDIT  
INTERACTION GRAPHICS CAPABILITY

FIGURE 2      EDIN OPERATING MODES.

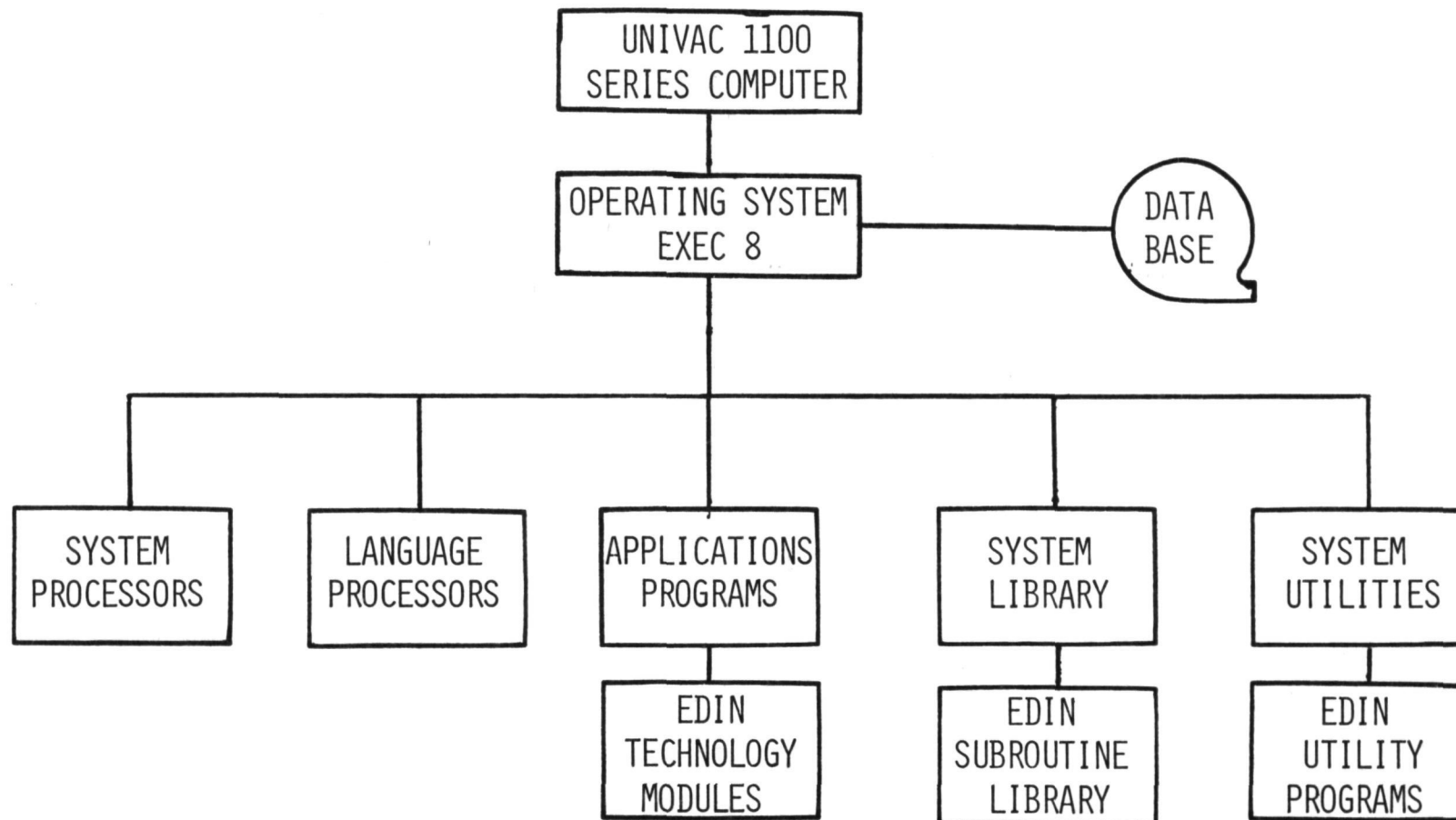


FIGURE 3 EDIN INTERFACE TO THE UNIVAC 1100 SERIES COMPUTER.

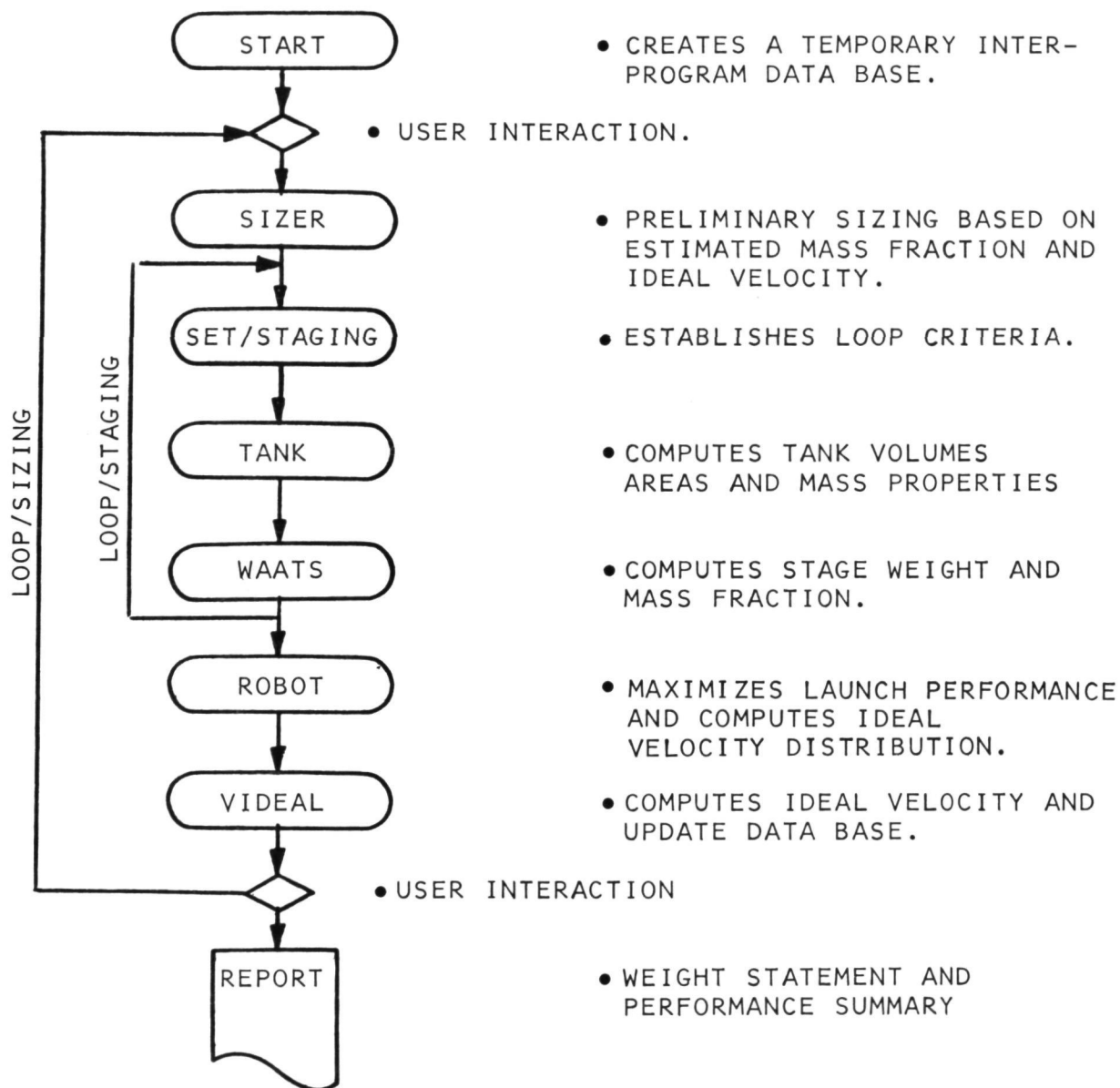


FIGURE 4 LAUNCH PERFORMANCE AND SIZING STUDY.

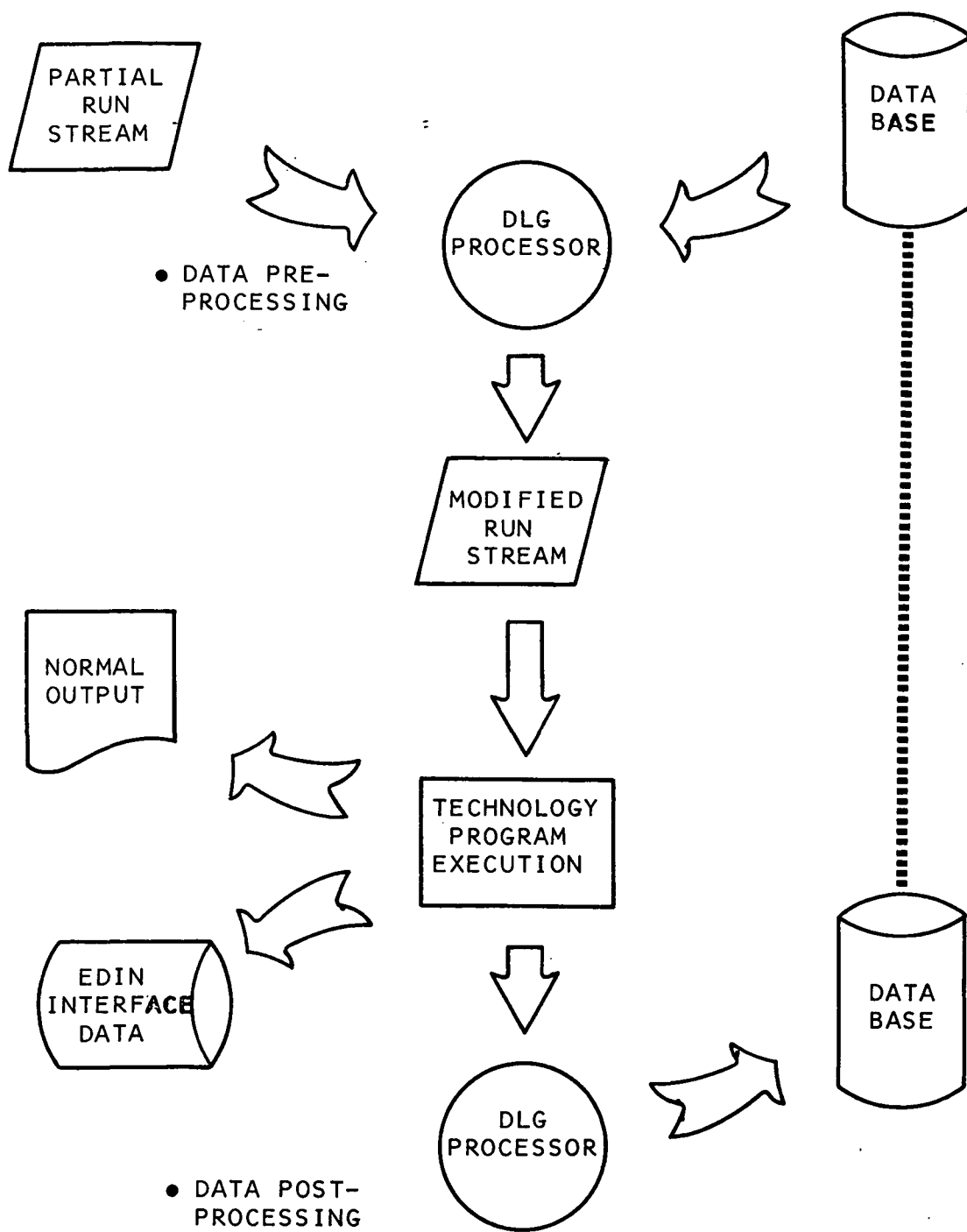


FIGURE 5 DATA PROCESSOR FUNCTIONS.

<u>GEOMETRY</u>	COLLECTION OF SPECIALIZED GEOMETRY GENERATORS.
<u>AERODYNAMICS</u>	EMPIRICAL SUBSONIC, TRANSONIC AND SUPERSONIC METHODS. LINEAR THEORY SUPERSONIC ESTIMATES. NEWTONIAN AND COMBINATION HYPERSONIC METHODS.
<u>PROPULSION</u>	ROCKET ENGINE SIZING AND THRUST CHAMBER DESIGN. TURBO/RAMJET ENGINE CYCLE AND INLET DESIGN.
<u>PERFORMANCE</u>	EIGHTEEN DEGREE-OF-FREEDOM MULTIPLE VEHICLE SIMULATION. GUIDANCE LAWS AND OPTIMAL STEERING/TARGETING. THREE DEGREE-OF-FREEDOM OPTIMIZATION BY PARAMETER OPTIMIZATION, FIRST AND SECOND ORDER VARIATIONAL TECHNIQUES.
<u>WEIGHTS</u>	HISTORICAL METHODS, PSEUDO STRUCTURAL ANALYSIS OF FUSELAGE STRUCTURES FOR SHUTTLE RELATED BOOSTER AND ORBITER CONCEPTS.
<u>STRUCTURES</u>	SIMPLIFIED AEROELASTIC LOADS AND FLUTTER. STRUCTURAL SIZING AND FINITE ELEMENT ANALYSIS.
<u>COST</u>	HISTORICAL METHODS BASED ON FIRST UNIT WEIGHT, DESIGN PHILOSOPHY AND OPERATIONAL PHILOSOPHY.
<u>BALANCE</u>	PREDICTIONS BASED ON CLOSED SHELL STRUCTURES AUGMENTED BY POINT MASS ANALYSIS.
<u>OPTIMIZATION</u>	PARAMETER OPTIMIZATION BY FIRST AND SECOND ORDER AND RANDOM PERTURBATION TECHNIQUES.
<u>GRAPHICS</u>	TWO AND THREE DIMENSIONAL DISPLAY GRAPHICS FOR ONLINE AND OFFLINE DEVICES. LIMITED INTERACTIVE PROGRAMS ON THE ADAGE 330.

FIGURE 6 EDIN SYSTEM SOFTWARE.

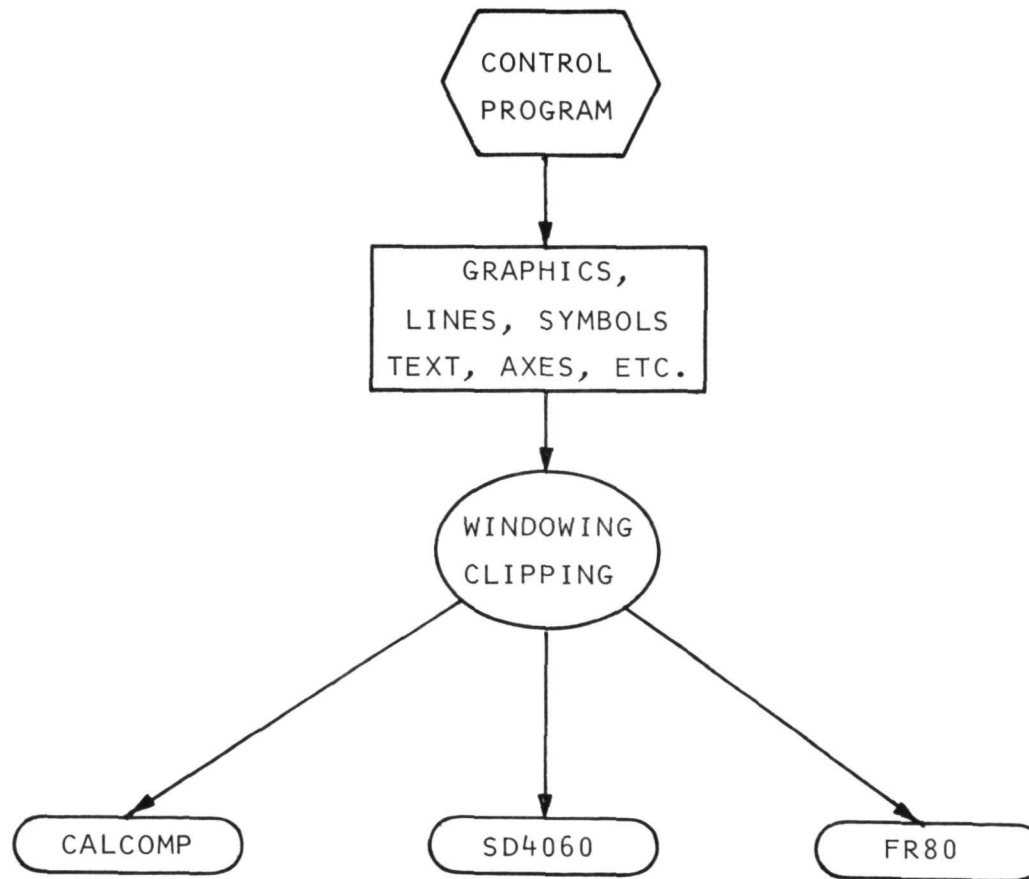


FIGURE 7 OFFLINE GRAPHICS.

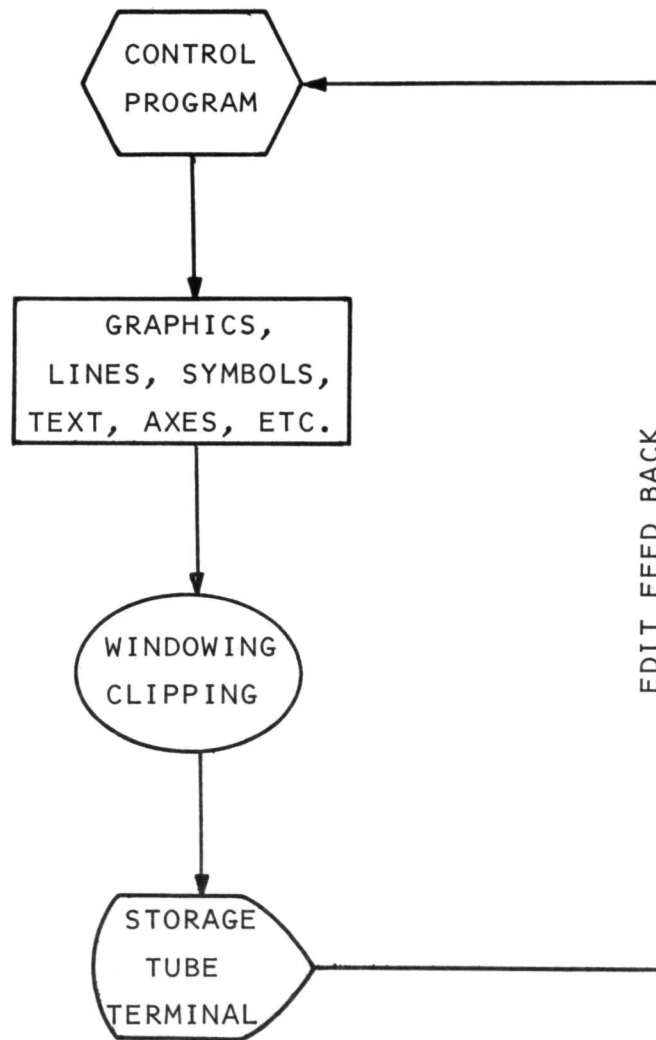


FIGURE 8 ONLINE GRAPHICS.



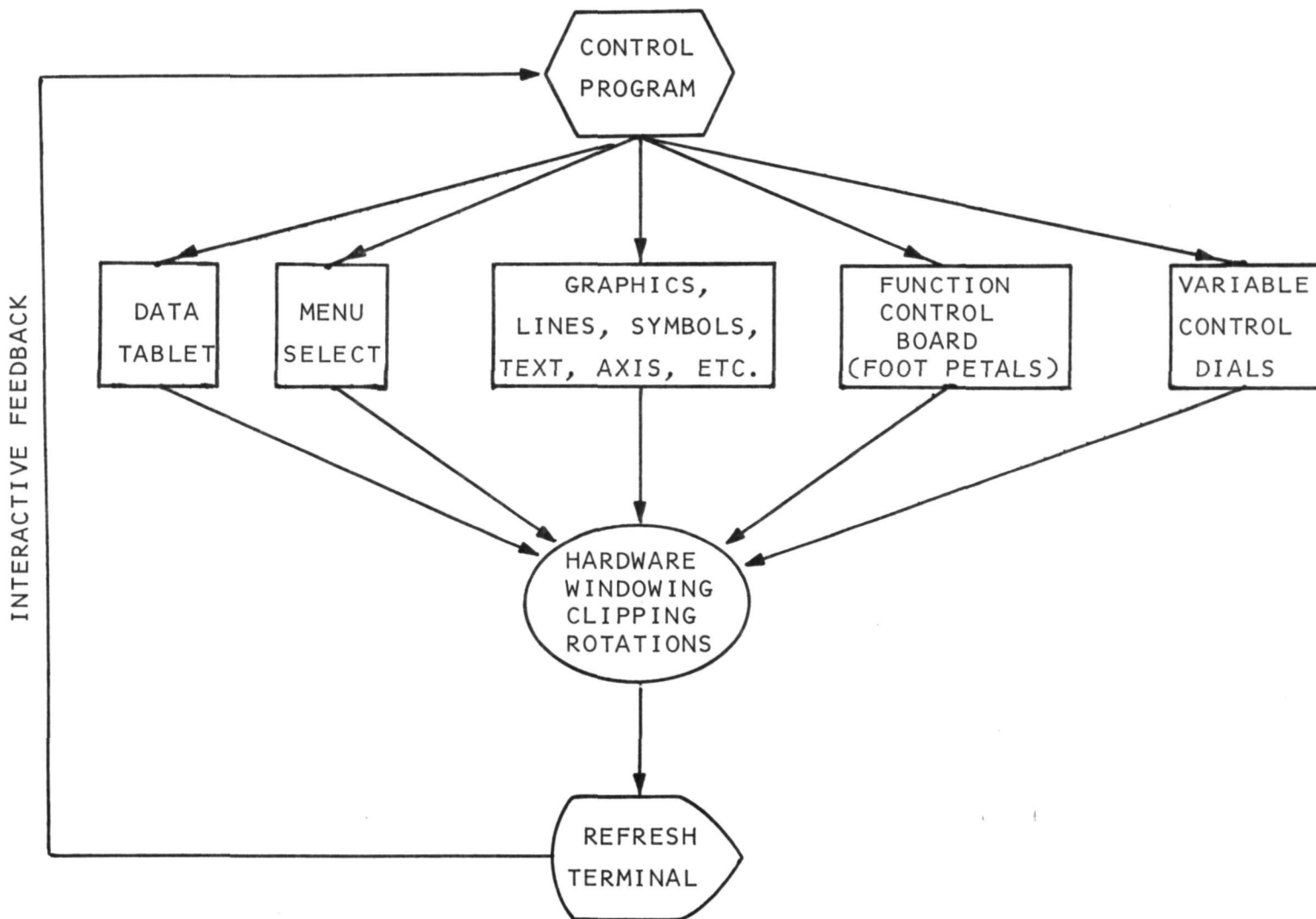


FIGURE 9 INTERACTIVE GRAPHICS.

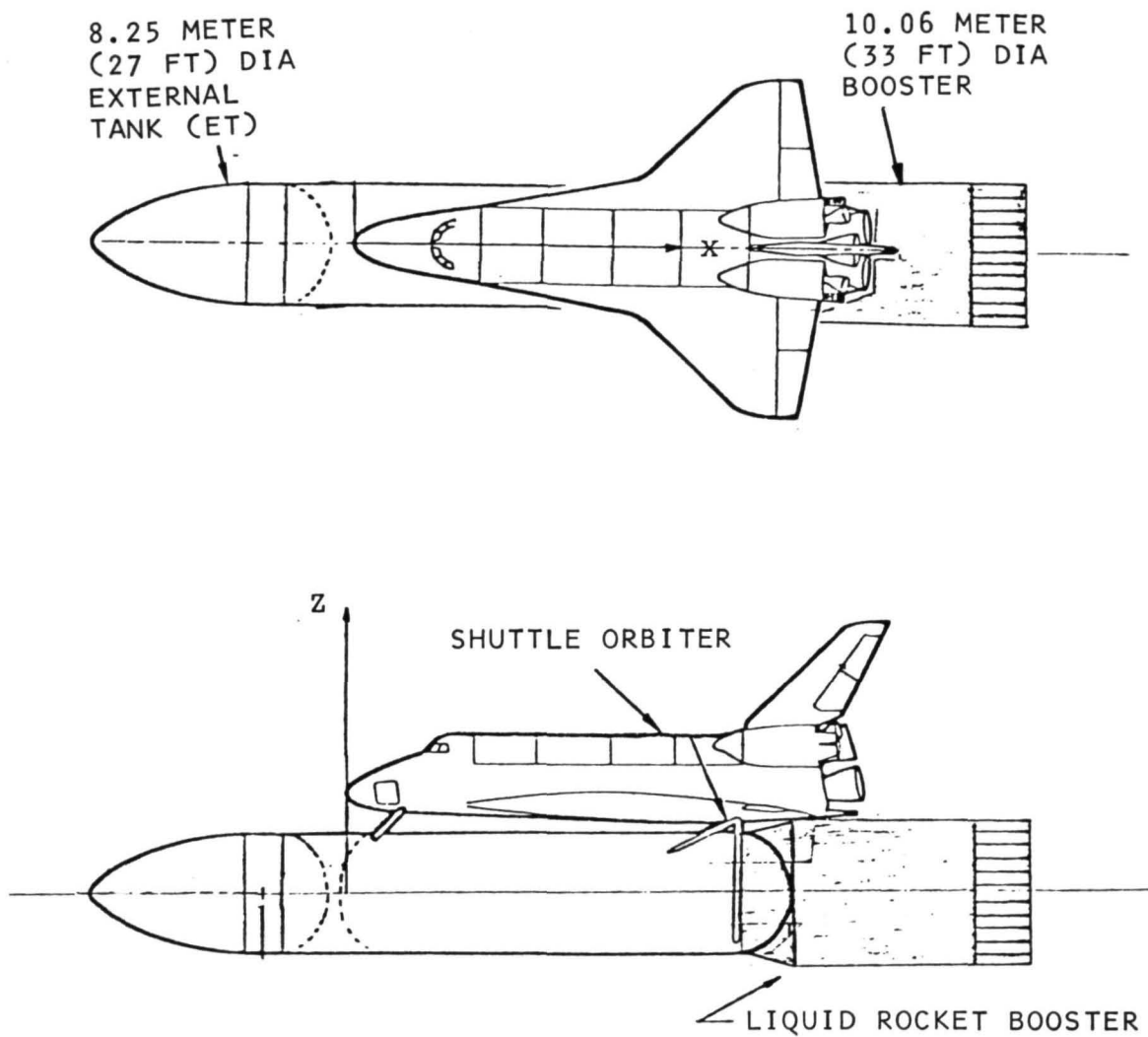


FIGURE 10 EDINO4 CONFIGURATION.

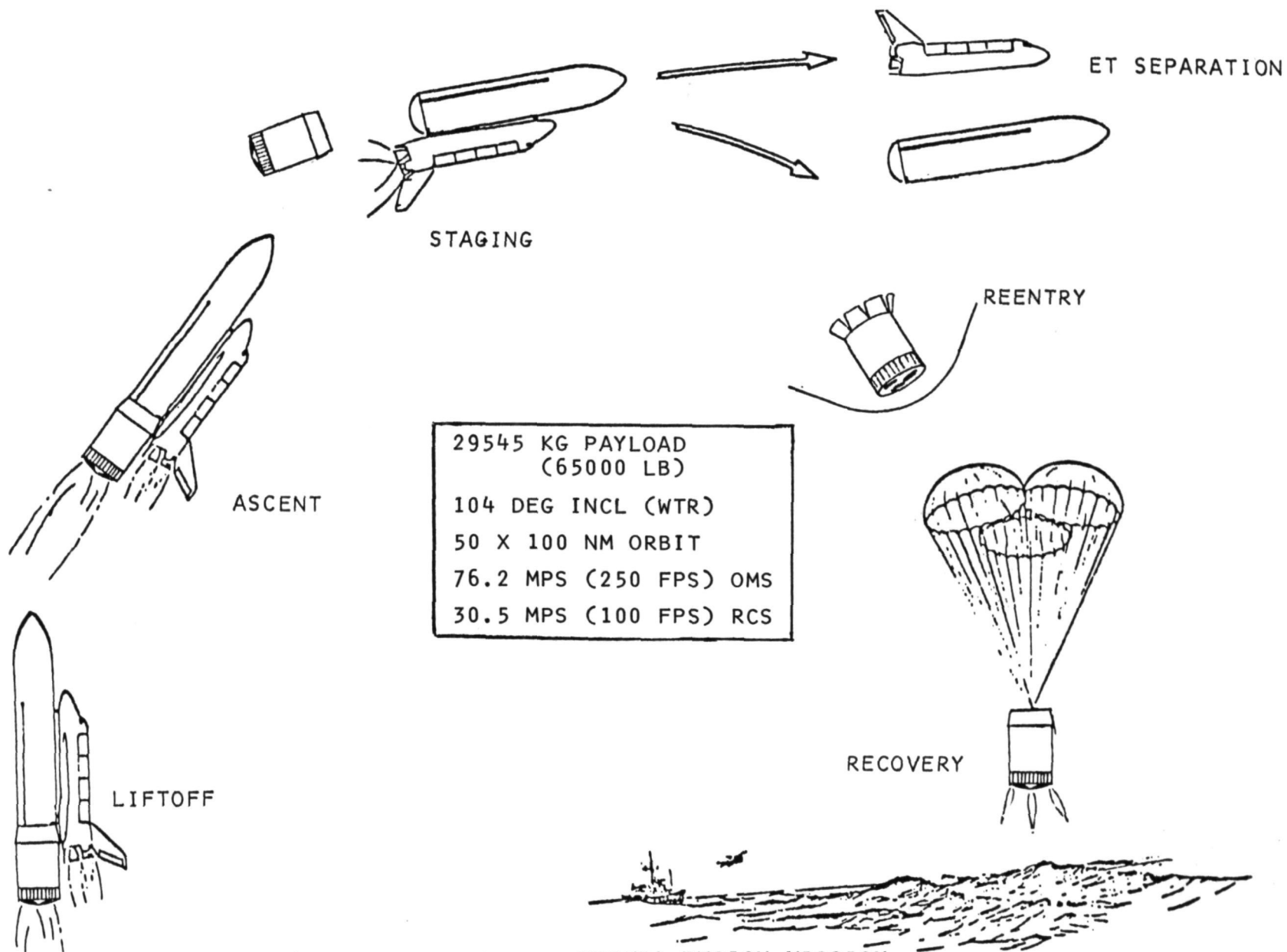
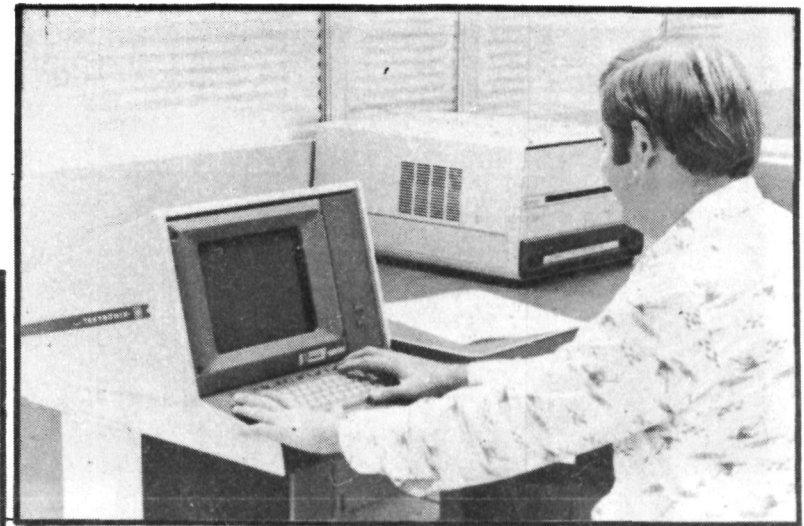
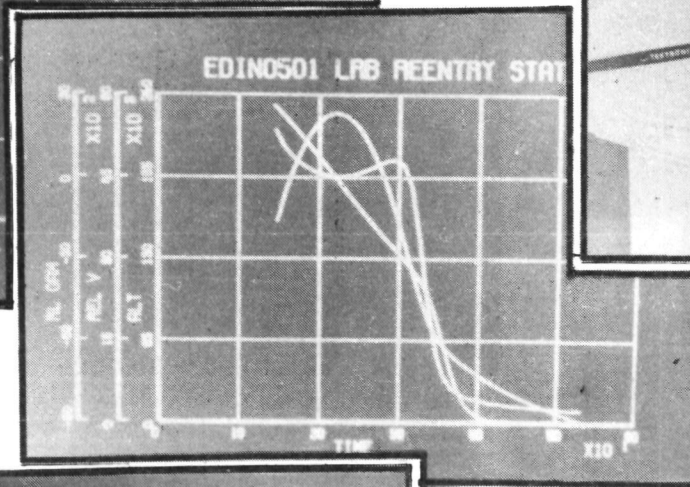
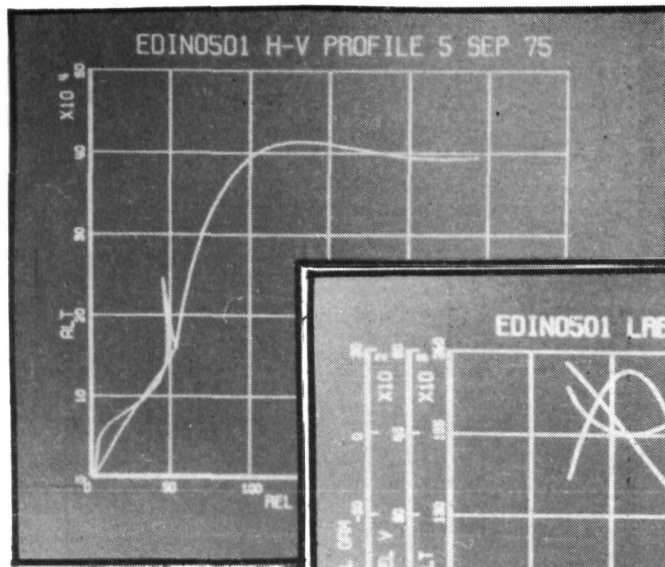
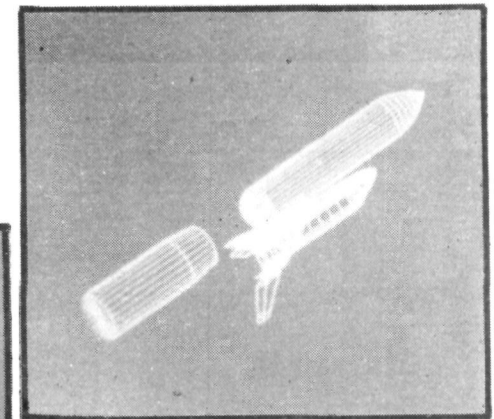
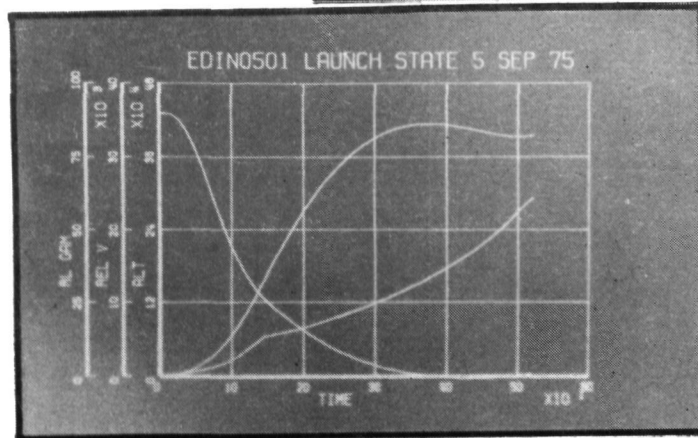


FIGURE 11

EDINO4 DESIGN MISSION



MISSION  
ANALYSIS



LRB  
SEPARATION

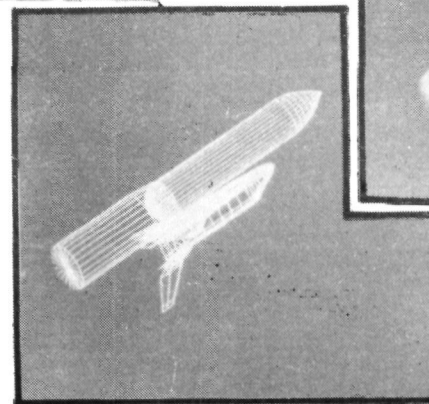
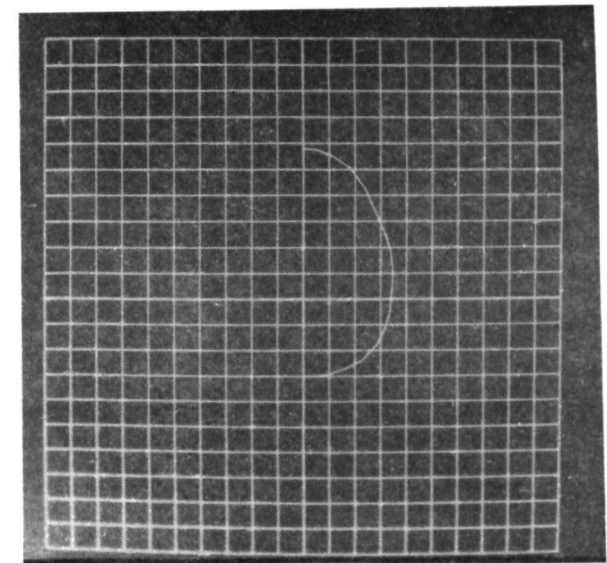
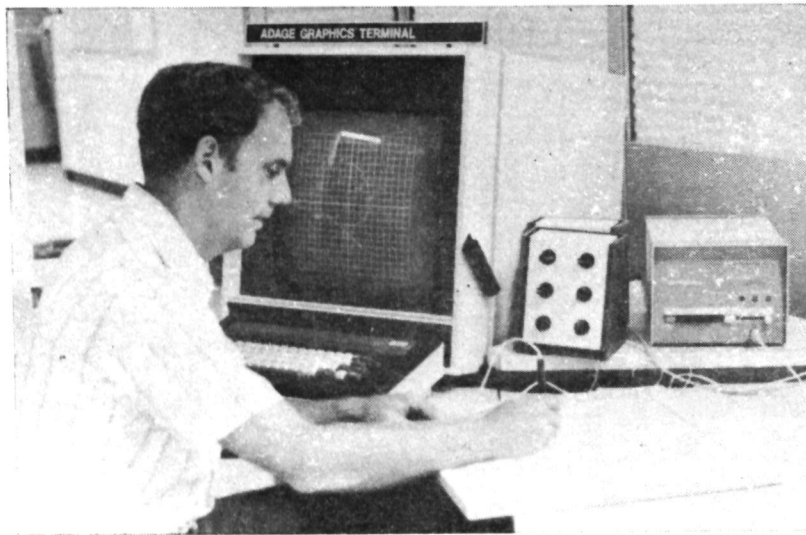
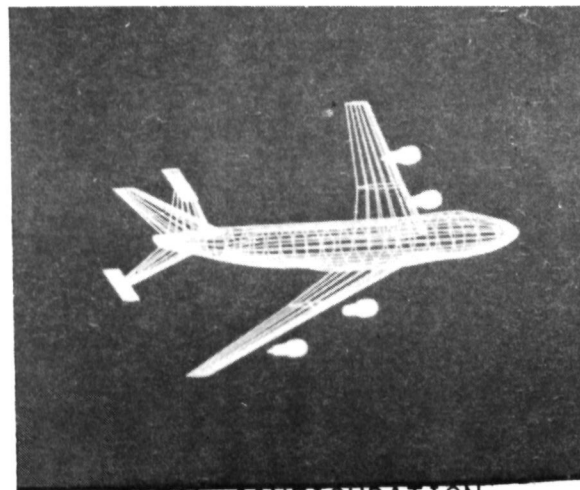


FIGURE 12

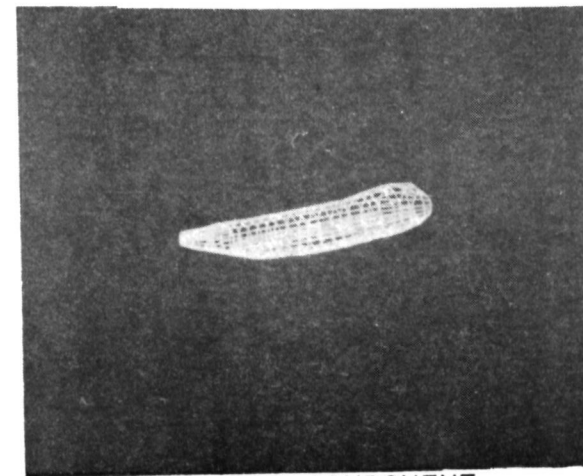
LOW COST GRAPHICS



FUSELAGE SECTION

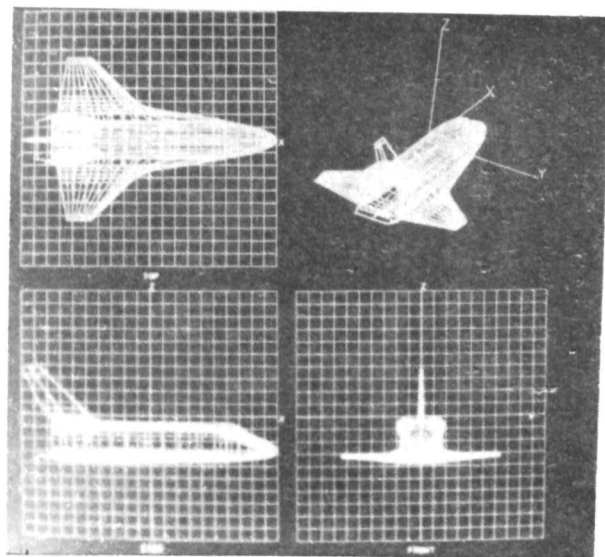


TOTAL CONFIGURATION

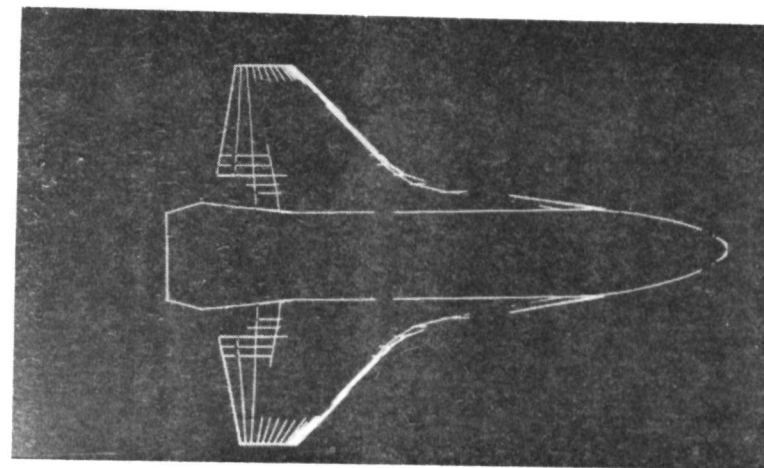


FUSELAGE COMPONENT

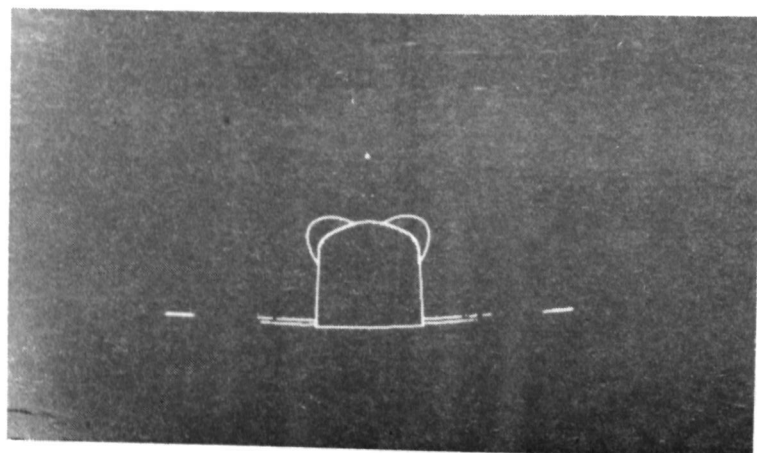
FIGURE 13 GEOMETRY INPUT MODULE



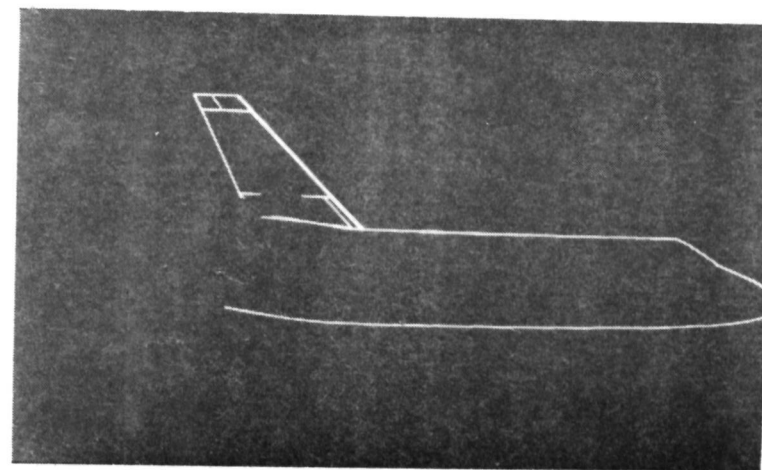
THREE-VIEW



PLANFORM



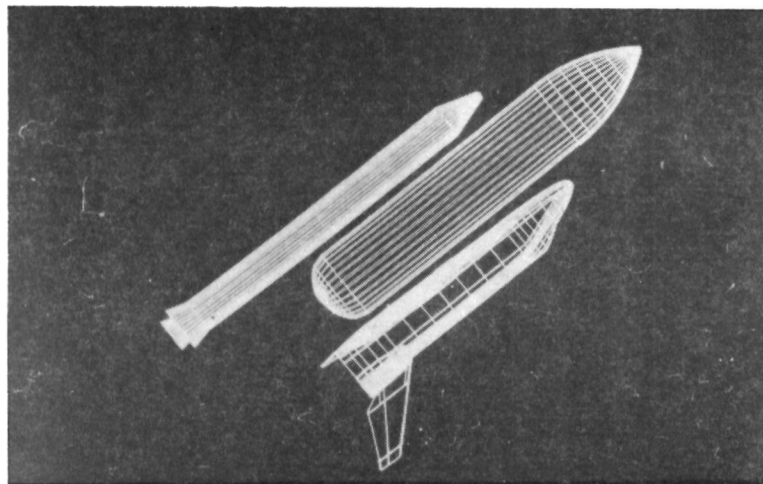
SECTION CUT



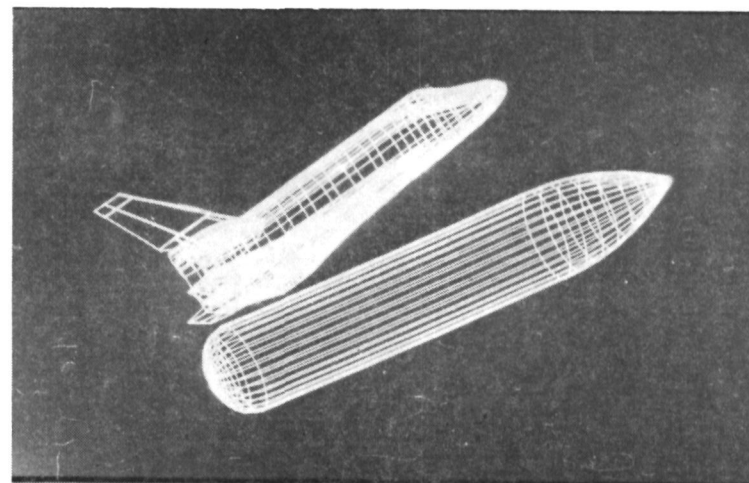
INBOARD PROFILE

FIGURE 14

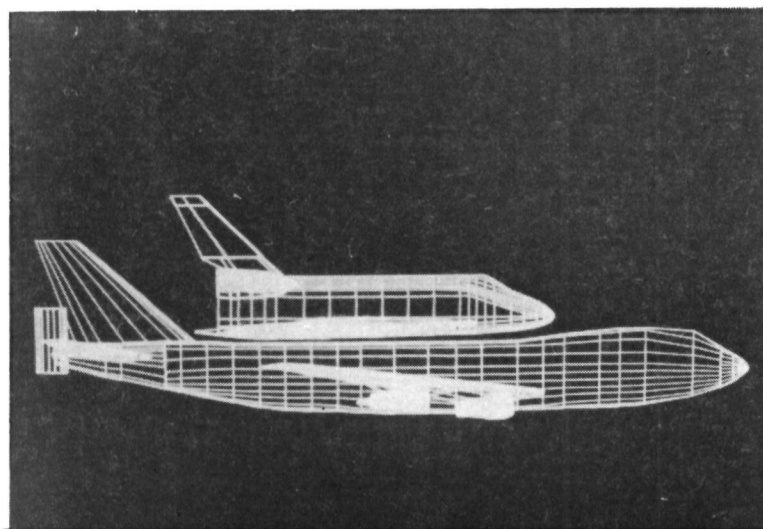
GEOMETRY EDITING MODULE



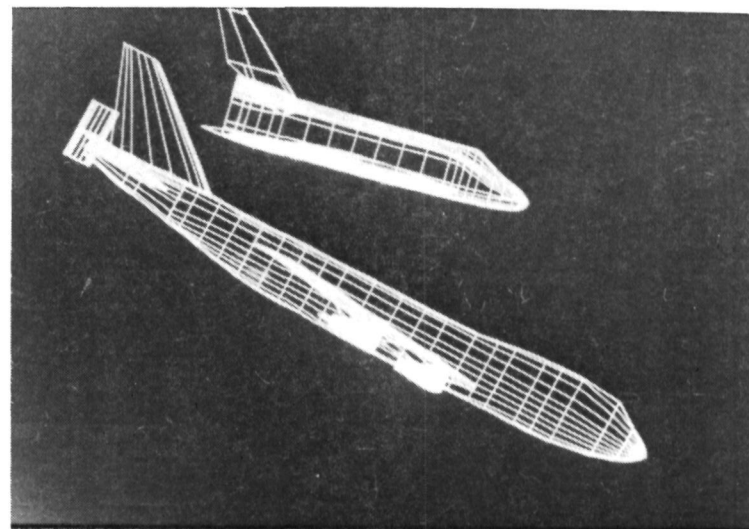
SRB BOOSTER SEPARATION



EXTERNAL TANK SEPARATION



SHUTTLE MATED TO BOEING 747



SHUTTLE SEPARATION

FIGURE 15 SEPARATION MODULE



SOME RESEARCH ADVANCES IN COMPUTER GRAPHICS  
THAT WILL ENHANCE APPLICATIONS TO ENGINEERING DESIGN

John J. Allan III  
Associate Professor of Mechanical Engineering  
and of Computer Sciences  
Director, Computer Applications Laboratory  
The University of Texas at Austin  
Austin, Texas 78712

SUMMARY

This paper describes some research in man/machine interaction, and graphics hardware/software that will enhance applications to engineering design. Research aspects of CAD executive systems, command languages, and networking are mentioned. Finally, a few areas where little or no research is being done are identified, and consideration is called for.

INTRODUCTION

This paper talks about research advances in computer graphics. This is not a paper on computer-aided design. The word research here means technical achievements that are not yet completed or commercially available. This paper is not intended to be exhaustive. In fact, it is really meant to stimulate thinking about things that have been and should be done. It is the author's feeling that more research needs to be done in computer graphics so that it can better and more economically enhance engineering design.

Further, these "advances" are discipline independent. Some of them relate to man/machine interaction, some relate to new hardware/software combinations, while still others refer to data structuring for graphic presentation, command languages for graphic input and output, and the design of executive systems and networks, especially as related to computer graphics.

MAN/MACHINE INTERACTION

Many of the things that we do when we design and build interactive graphic displays for use in engineering design are done by what we call "gut feel." This is fine, and in many cases it may even be accurate. However, the increasing sophistication of both the CAD system behind the graphic display, and indeed, the graphic display itself, demands that we have solid data on which to base the decisions about the design of the software and hardware for our displays.



Consequently, work has been done to substantiate, or disprove, certain ideas or concepts that many engineering design practitioners have had about interactive graphics. In fact, this paper is, in part, a call for a much greater effort in this area. The reason for that is that in the not-too-distant future, as our colleagues in artificial intelligence and pattern recognition make greater and greater advances, and as computing equipment becomes even less expensive, it will be possible to have CAD systems which adapt to the individual design engineer. When this comes about, it is going to be extremely important for us to be able to design those systems so that when they adapt to a new designer, they adapt in a most effective way. That will give the designer the greatest freedom for making decisions in the design process.

One study completed in 1971 and reported in 1973 (1)\* by Corley, reported for the first time a statistically significant experiment in which a large group of design engineers were asked to route some piping through a factory such that the thermal-stresses would be minimized. These engineers were asked to work on problems of four levels of complexity. Furthermore, each one was asked to route the pipe through the plant topologically in a keyboard mode, in a drawing mode through a tablet in which the drawing was drawn to scale, and then in a third mode, which was a combination of these in which the pipe-routing topology was entered by drawing but the dimensions were entered by keyboard. The point was to verify the fact that topologically entering the pipe-routing by drawing is not only a significantly quicker way to enter the data than by keyboard, but is less error prone. Also, it was desired to investigate the learning rates of these design engineers as they went to consecutively tougher problems using this new equipment. Their learning rates did show up in the data.

In subsequent and more substantial pieces of work, Corley reports using pragmatic information content as an aid in the design of interactive graphic displays.

In one report Corley (2) reports on the specification of the pragmatic information content of graphical displays as approached by defining three display attributes: quantity, format, and time. He proposes a mathematical measure of the information quantity in the display, and discusses qualitatively the effects of format and time on pragmatic information processing. He presents data from an experimental evaluation of the effects of quantity, format, and time on pragmatic information processing in a specially constructed graphically-oriented engineering task.

In another publication, Corley (3) develops the pragmatic information content of a "frame" in an interactive graphic display. He also presents in that publication an example of cam design, and goes into detail as to the predicted quantity of information in the display.

---

\* Parentheses refer to numbered references at the end of the paper.

In another study, Beazley (4) discusses the evaluation of interactive graphics used in the engineering design environment. He presents a general theory of design behavior, and the essential elements of design communication are then reduced to the study of digraph messages. The study relates to the construction and editing of models of engineering systems done by selecting nodes and specifying their relationship to each other with arcs. He presents the experimental results of a study of digraphs, their formats, and the associated learning tasks. This is all presented as a study of computer graphics acceptability for engineering design use.

In the Beazley study the formats of the screen were chosen to evaluate the special features of interactive graphics which distinguish it from highly non-spatial alpha/numeric terminals. The results indicated that digraphs could be learned with significantly less learning errors when the subjects were permitted to use the spatial cues available in interactive graphics. Other results are included and discussed.

A current study is underway which will use a group of design engineers to evaluate the effect of audio feedback on problems in which one of the cues to the designer in the real world would be audio feedback. This is applicable to many problems such as tolerances in mechanical linkages, the mounting of loud speakers in audio equipment, and for the particular problem chosen in this study, the design of an acoustically treated computer-aided design laboratory. Subjects will be asked to design an acoustically treated CAD laboratory to optimize the noise levels for the design engineers that will be working there. As the experimental subjects sit at the CAD terminal designing the various configurations of the room, the D-A equipment will be driving noise generators, which will feed back to them audio signals simulating the noise level of the room that they are designing. Control groups are being held that are not being given the audio feedback so that the effect of the audio feedback in achieving the design criteria can be measured.

It is in this author's future plans, when the equipment becomes available, to run controlled studies of audio input coupled with interactive graphics. A first paper in this area was presented by Griffith and Riganati (5) in which they discuss an interactive graphics system with bi-directional audio capability. It has been used for classifying and describing speech and image data. While not slanted specifically to engineering design, it has been used extensively by signal processing and pattern recognition groups to perform research and advance development studies in speech, image, specialized low-cost optical character recognition, and fingerprint processing.

#### NEW HARDWARE/SOFTWARE DEVELOPMENTS

The video frame buffer and some of its associated software as described by Kajiya, and Catmull (6,7) is a new computer graphics facility for displaying three-dimensional objects. Probably as important, or more important than the hardware frame buffer itself and its

associated technological accomplishments, is the thinking that has been going on about the way to store data which can subsequently be transformed to represent three-dimensional objects. The papers referenced here describe a random-access semi-conductor memory and discuss the central design considerations necessary to drive a shaded picture television display. Also presented in these references is a method for producing computer-shaded pictures of curved surfaces, including the notion of mapping onto patches of those surfaces, texture for computer-generated pictures.

In other publications Laws, and Sproull and Newman (8,9) describe gray-scale graphic processing from both the hardware and software points of view on raster-scan graphic displays. These papers explain the main reasons for choosing to build a run-code gray-scale graphic processor, and indicate some of the problems in the design. Also included is the description of a graphics package for use with raster-scan displays. This latter involves an extension to conventional techniques for constructing general-purpose line-drawing graphics software for the definition and display of solid opaque objects.

In a paper by Denes (10), he describes a computer-controlled scan display system specially designed for interactive applications. A particular feature of the system is that the color of each display element is under program control.

Because high-speed interaction with a conventional refresh graphic system is most desirable, this author and one of his colleagues have just installed 16-bit parallel 50-kilobaud interfaces from a PDP-11/40 to their four Imlacs. This hardware is in part described by Stowell (11) in an article in which he describes the architecture of an interactive three-dimensional random vector display processor. Some of the real-time capabilities provided by the hardware described in the paper are picture rotation, translation, scaling, perspective projection, manual picture manipulation, programmable intensities and line-types, selective editing, mirror imaging, and windowing.

Other graphic devices such as the laser-scan display developed at Cambridge are all contributing to the further enhancement of the picture that can be presented to the design engineer.

The point of including the above references to new hardware and software is that while they may be argued out of the picture for economical engineering design in 1975, they are not as drastic a departure from what we are doing today, as Sketch-Pad was, from what we were doing in 1963. The point here is that it is incumbent upon engineering designers and CAD professionals to learn how to best take advantage of the computer graphics capabilities that can act as a most natural front-end to a computer-based engineering design system. It appears that the real hang-up is how to structure the data. Regardless of whether the data are going to be handled by hardware or software, how to structure them is not adequately known.

To that end, Allan (12) calls for further work on formal data-base descriptions so that transformers that yield graphic post conditions can be applied. It is well-known that engineering designers would like to be able to: (1) free-hand sketch for "feel," (2) prescribe the topology of a reticulated physical system, (3) enter complex surfaces that describe physical property data, (4) handle imprecise or qualitative data, (5) connote "solidness," (6) plot 3-D physical objects with gray-scales, (7) provide output animation, (8) "recognize" digitized pictures of prior designs, (9) etc. It is trying to get all of these representations from the same data that is forcing the need for new and better graphics hardware, software, and data structuring techniques.

#### OTHER RELEVANT CONCERNS

Three other major relevant concerns have a bearing on computer graphics as related to engineering design. Developments in CAD executive systems, in computer graphic command languages, and in computer graphic networking are all relevant areas.

The importance of CAD executive systems, as they relate to research advances in computer graphics that will enhance applications to engineering design, is that in the overall design of a CAD system the type of graphics and the use that it will be put to has to be integrated with the design and capability of the entire CAD system. Further, this must be considered in terms of the environment in which it is going to be used. So to this end, a great deal of research needs to be done on functioning CAD systems to find out the relative effectiveness with which types of graphics devices and the methods in which their messages are formatted can increase the effectiveness of the design engineer.

The area of command languages is of vital importance because being able to couple the type of computer graphic facilities with the appropriate ability to communicate concepts and ideas will give rise to the more cost-effective utilization of computer graphics in engineering design. This will continue to be a problem as the latest graphics equipment available is much more expensive than what is called conventional engineering design graphics as generally reported in this conference.

It is apparent to most practitioners in CAD that it is going to soon become economically unviable for everyone to generate his own software. Therefore, parts and pieces of software are going to have to be shared from distant locations. Because the type of graphic device dictates the protocol for bringing in messages, the relationship to networking is going to be important. That fact is prompting the IIASA/IFIP W.G. 5.2 task group on International CAD Networking to look into standards,

protocols, etc. A paper in this area by Cohen (13) discusses the network issues of computer graphics over a general purpose digital computer communication network. It is pointed out in the article that the huge variety of graphics equipment and general purpose computers is the real problem to solve. The network transmission issues are well-understood and generally solved. It goes on to point out that non-trivial effects of a network on graphics are band width, error-control, and delay issues. This will be a particularly greater problem as corporations and government institutions with diverse design groups want to begin to communicate on an integrated CAD system coupled by networking. How we design graphics and how we use graphics for engineering design in that environment will take a great deal more research.

### NOW AND IN THE FUTURE

When Hatvany, Newman, and Sabin (14) were commissioned by IIAS to do a world survey of computer-aided design, one of the areas to which they paid special attention was computer graphics. The reason that this paper argues that a great deal of research needs to be done in the area of computer graphics is that we need to turn computer power on itself in the area of applications much as the manufacturers have done in the area of manufacturing. We need to be able to use higher-level recognition of symbols, concepts, and ideas in order to make the conceptual input of computer graphics as applied to engineering design realizable. The argument that is always used against us is that computer graphics is too expensive. To quote from the world survey of computer-aided design: "Overall, our survey of the use of computer graphics in CAD leaves us to make the following observations: (1) we have found a great interest in 'turn-key' graphics systems, particularly among the electronics firms. These systems enable the user to ignore the programmer's problems and also permit the design of graphical 'front-ends' to existing batch operated programs. (2) Computer graphic equipment is expensive to buy and expensive to use. These are less serious problems now than they once were, but we will see great hesitation to use graphics in CAD because of the cost. (3) Designers of graphic software are reluctant to try new techniques partly because of ignorance and partly because of programming difficulty. This prevents economic use of computer graphics. Techniques we would like to see more widely used are procedure-driven rather than data-structure-driven graphic output, display files used for the update of storage tube displays, and on-line character recognition as an input technique. (4) Graphic hardware is rarely designed with the programmer in mind and graphic system hardware shows singular lack of understanding of the application programmers needs. There is a need for simpler hardware and more flexible software. We have seen signs of progress in these directions. (5) It is appropriate to consider drawing up standards for the functions to be provided by graphics packages. IIASA should maintain contact with the SIGGRAPH graphic standards planning committee with a view to benefitting for many standards or definitions that this committee produces."



In light of the last section of their observations, IFIP W.G. 5.2 has also developed a committee which is in liaison with the SIGGRAPH graphics standards planning committee to develop a set of graphics standards that will probably reflect to a great extent those things found in GINO-F and other similar graphics packages.

Another area of graphics research has to do with the social ramifications of using computer graphics in the area of engineering design. Some of the many questions that we should address ourselves to relate to what happens when we use computer graphics to increase the effectiveness of engineering designing systems. If we displace people in their communities, if we give increased leverage to large industries that can afford sophisticated CAD systems, if we spawn a group of small specialist industries, or if we subsequently lose certain mid-size concerns, what advance planning should we do?

Another aspect that we should consider in the area of engineering design is: what should be changed in education as computer graphics is proliferated and becomes evermore economically viable? On the other hand, there are questions which are even further out in the future. For instance, computer graphics gives us a new way to communicate to analysis programs like we have never had before. And as computer graphics becomes more viable, and is coupled to even larger-scale computation, it may be that we will be able to design products that the average person cannot comprehend. What is the social consequence of that action?

#### CLOSING REMARKS

The whole idea of research into computer graphics as it is related to engineering design comes back to the fact that we are in a technologically-based society. For that reason, to keep improving our lot, we design what we collectively feel are new and better things. Because of the competition of free enterprise and other pressures, we would like to design ever more comprehensively and ever-faster. This has driven us to find a syntax with which we can convey ideas, concepts, and trends without resorting to some complicated syntax to communicate with large data bases and large computational schema. The drive for this is economics. The resistance in the engineering community to computer graphics is purely economic, which includes the educational aspect. So, from that point of view, we need to plan wisely and do research so that we are not always flying just on the "gut feel" that we have about the goodness or the badness of graphics. To that extent this paper then is a call for more studies on man/machine interaction and a call for cooperation in the integration of the latest in graphics hardware and software into a pragmatic CAD environment so that the effects of the new hardware can be measured. All the time, this should be done by people who realize that they are responsible professionals in a total environment and therefore will also incorporate research into the social implications of our ability to communicate ever-faster and ever more comprehensively with engineering design systems.

## REFERENCES

1. Corley, Melvin R., and John J. Allan III; "The Effectiveness of Direct Graphical Entry of Topological and Geometric Data," Journal of Behavior Research Methods and Instrumentation, Vol. 5, No. 2, 1973, pp. 197-199.
2. Corley, M. R., and John J. Allan III; "Pragmatic Information Processing Aspects of Graphically-Accessed Computer-Aided Design," Proceedings of IEEE 1975 International Conference on Cybernetics and Society, WeAM4, September 23-25, 1975, San Francisco.
3. Corley, M. R.; "Use of a Pragmatic Information Measure as an Aid in the Design of Interactive Graphic Displays," ASME Publication 75-DET-88, 1975.
4. Beazley, William G.; "Man-Machine Communication of the Structure of Engineering Design Problem Information," MS Thesis, Department of Mechanical Engineering, The University of Texas at Austin, 1974.

References 5 through 13 are all to be found in Proceedings of the Conference on Computer Graphics, Pattern Recognition, and Data Structure, IEEE #75CH0981-1C, May 14-16, 1975.

5. Griffith, M. L. and J. P. Riganati; "Interactive Audio-Graphics for Speech and Image Characterization," pp. 163-169.
6. Kajiya, James T., Ivan E. Sutherland, and Edward C. Cheadle; "A Random-Access Video Frame Buffer," pp. 1-6.
7. Catmull, Edwin; "Computer Display of Curved Surfaces," pp. 11-17.
8. Laws, B. A.; "A Gray-Scale Graphic Processor Using Run-Length Encoding," pp. 7-10.
9. Sproull, Robert F., and William M. Newman; "The Design of Gray-Scale Graphics Software," pp. 18-20.
10. Denes, Peter B.; "A Scan-Type Graphics System for Interactive Computing," pp. 21-24.
11. Stowell, Garrett W., and Richard E. Garrett; "A Three-Dimensional Display Processor Design," pp. 157-162.

12. Allan, John J.; "It's Time for a Different View of Data Structures for Computer Graphics," pp. 309-311.
13. Cohen, Dan; "Computer Graphics Over Computer Networks," pp. 294-296.
14. Hatvany, Joseph, William M. Newman, and Malcolm Sabin; Report to IIASA on a World Survey of Computer-Aided Design, International Institute for Applied Systems Analysis, Laxenburg, Austria, July 1974, 52 pages.



**Page Intentionally Left Blank**

AN INTERACTIVE GRAPHICS PROGRAM TO RETRIEVE,  
DISPLAY, COMPARE, MANIPULATE, CURVE FIT,  
DIFFERENCE AND CROSS PLOT WIND TUNNEL DATA

Robert D. Elliott, Norbert M. Werner,  
and William M. Baker

Lockheed California Company

SUMMARY

The Aerodynamic Data Analysis and Integration System (ADAIS), developed at the Lockheed California Company, is a highly interactive computer graphics program capable of manipulating large quantities of data such that addressable elements of a data base can be called up for graphic display, compared, curve fit, stored, retrieved, differenced, cross plotted, and output for hardcopy plots. While the principle use of ADAIS thus far has been in the aerodynamic analysis of wind tunnel model force and moment data, the general nature of the system and its potential for broader application is evidenced by the fact that limited usage has already occurred with data bases consisting of thermodynamic, basic loads, and flight dynamics data. Productivity using ADAIS of five times that for conventional manual methods of wind tunnel data analysis is routinely achieved.

In using ADAIS for wind tunnel data analysis, data from one or more runs of a particular test may be called up and displayed along with data from one or more runs of a different test. By means of light pen detection, the principle mechanism of interaction with the program, the parameters for ordinate and abscissa are selected. Scaling of the axes to fit the data range is automatic, with an option for manual over-ride via the typewriter keyboard. Curves may be faired through the data points by any of four methods, including cubic spline and least squares polynomial fit up to seventh order. Improvement to visually unacceptable curve fits can be accomplished by deletion of certain data points from the curve fit while continuing to display the deleted points.

Another option permits weighting of selected data points by moving them up, down, right or left incrementally for curve fit purposes while observing the effect of each movement on the fairing with respect to the original data point positions. With practice, curve fits as good as those obtained manually can be achieved in a very short time, due to the interactive nature of the process.

Once a family of curves is defined mathematically by a visually acceptable fairing, increments between curves can be obtained, graphically displayed, and faired in the above manner. Each of these incremented curves may then be given an identifying numerical value corresponding to some physical parameter of the test. Upon specification of a few additional control items, a cross plot can then be obtained.

A request for hardcopy print, accomplished by light pen detection, results in generation of a magnetic tape from which a 35 mm negative is produced in an off line process. Hardcopy is made from the negative by an electrostatic process of photo enlargement. Grid spacing is one millimeter on metric plots and major and minor accented lines are shown, just as in commercially printed graph paper. Hardcopy quality is adequate for use in published reports and has been so used.

The data base is resident on a Disc Pack and occupies approximately 14 million bytes of storage. Presently the data base consists of five major data sets. Each data set includes an average of 120 runs. Each run consists of an average of 18 test points. For each test point there is provision for storing 12 elemental items of data such as lift coefficient, drag coefficient, etc. The result is an addressable data base of over 2.4 million elemental items of data with space available for up to 3.5 million items total.

Adding a new test to the data base is routinely accomplished within 24 hours of receipt of the wind tunnel test data tape and has been accomplished in as little as three hours elapsed time. Older tests are removed from the data base to off line storage from time to time to make way for new ones.

## INTRODUCTION

In conventional aerodynamic analysis of wind tunnel test data, both tabulated and automatically plotted aerodynamic coefficient data are available. Usually a magnetic tape or punched cards containing the same data is also available at the conclusion of the test. In a well run wind tunnel test, where a plotting schedule has been adequately pre-planned, automatically plotted data, available at the end or shortly after the end of the test, present different test runs logically grouped on particular plots so as to facilitate subsequent analysis. Faired curves through the data points are usually not included in these batch-generated plots, and when they are included they are often unacceptable to the aerodynamicist. Therefore, only symbolized data points are usually presented and curve fairing is accomplished manually on ozalid or vellum plots of the data.

The bulk of wind tunnel data analysis deals with evaluation of increments between runs rather than absolute levels of aerodynamic coefficients. Furthermore, there are almost always comparisons to be made with previous tests made with the same or slightly different models, possibly in a different tunnel. Therein lies a severe logistics problem. Previous test data are usually found in large bound volumes of 11 by 17 inch ozalid prints of plotted data. When these are located, the appropriate runs selected, and the corresponding plots located and removed from the bound volume, it is frequently discovered, in overlaying of old and new test plots on a light box, that differences in graph paper and stretch in the reproduction process are sufficiently large to preclude this method for assessing small differences between runs. What is almost always resorted to is a manual replotting of tabulated data from the separate tests on a clean piece of graph paper, a time consuming process.

Even for that portion of the analysis that is done within a single test, considerable hand plotting is unavoidable. Suppose it is desired to find the variation of incremental drag coefficient due to spoilers versus spoiler deflection at several specified values of angle of attack. Available is a batch-plotted display of data points of total drag coefficient versus angle of attack for several runs, each at a different spoiler deflection and including one at zero deflection. After fairing curves through the data points, increments between the zero deflection curve and each of the other curves must be developed and plotted as incremental drag versus angle of attack on a separate plot. This usually involves use of a pair of dividers. Once this intermediate plot is faired, the data can be manually cross plotted using dividers again, or manually digitized and replotted to obtain the desired final plot.

In 1970, Michael I. Grove, then of LCC's Commercial Engineering Aerodynamics, recognizing both the need to speed up the storage, retrieval, and analysis of wind tunnel data and the availability of computer graphics equipment, conceived an Aerodynamic Data Analysis and Integration System (ADAIS) to make use of available computer graphics equipment. Programming was initiated in June 1970 and by November 1970, ADAIS was developed to the point of productive use. By July 1971, at which point about two man years of programming effort had been expended, significant production use had been accumulated. An additional man year of effort was expended between July 1971 and June 1975 in continued improvement of the system.

The relatively general nature of the ADAIS system and its potential for wide application is evidenced by the fact that it has already been used on the following problems other than aerodynamic wind tunnel data:

- o Loads - Pressure distribution data set up in a data base, accessed, manipulated, and faired.
- o Flight Dynamics - Bode plot data for numerous servo system variations were stored, retrieved, compared, and plotted.
- o Thermodynamics - Using a data base consisting of batch program output of the variation of several thermodynamic system variables along an aircraft flight profile, selected variables such as temperature, inlet area, and ram airflow, were plotted versus altitude for the L-1011 Environmental Control System. Extensive use was also made of the zoom in/zoom out feature to "zero in on" and blow up certain areas of the flight profile history which were of particular interest.

#### NOTATION

ADAIS	Aerodynamic Data Analysis and Integration System
ALPH	Angle of Attack, Alpha
CADAM	Computer Augmented Design and Manufacturing
CD	Drag Coefficient
CPU	Central Processing Unit
DSP	Incremental Spoiler Deflection
IBM	International Business Machines
INCR	Incrementation
LCC	Lockheed California Co.
LPD	Light Pen Detect
STC	Storage Technology Corporation
<b>VG</b>	<b>Vector General</b>

## COMPUTER ENVIRONMENT

The principal scientific computer complex at the Lockheed-California Company (LCC) is built around an IBM 360/91, shown in simplified schematic in figure 1. The three methods of communication with the computer are 1) graphics, which is direct access, on line, and interactive, 2) other direct access devices limited to remote job entry (RJE) and program and data edit, and 3) normal batch, usually input via punched cards.

Most graphic work is accomplished during day shift when the two million byte core of the IBM 360/91 is partitioned as shown in figure 2. Also shown are the relative priorities for access to the Central Processing Unit (CPU). ADAIS operates as an analytic graphics program which is third in access priority to the CPU and where each partition is devoted to only one graphic scope. Response times are typically under one half second which is usually faster than the user can supply the next instruction.

In addition to the digital computer, other hardware is necessary to process a magnetic tape to obtain 35 mm microfilm from which hardcopy plots are obtained by an electrostatic process. The entire process is shown schematically in figure 3.

Graphics consoles from two manufacturers, IBM and Vector General, are used interchangeably at LCC and are pictured in figures 4 and 5, respectively. Of the three means of direct communication with the computer: 1) light pen, 2) function keyboard, and 3) typewriter keyboard, the light pen is the principal one used with ADAIS. ADAIS also makes use of 17 function buttons on the function keyboard, as indicated in the function keyboard overlay template shown in figure 6.

The ADAIS source deck consists of about six boxes of cards containing 48 subroutines. Because it must operate in a 130K partition, it is tightly overlaid. Without overlay the program would occupy approximately 230K bytes of computer core. The main logic flow, shown in figure 7, gives an indication of the many program elements which must be interchanged within the partition.

## INTERACTIVE GRAPHIC UTILIZATION

The interactive use of ADAIS is explained in this section via a step-by-step treatment of a typical session at the graphics terminal. The session was designed to illustrate most of the main features available, such as combining data from different tests on the same plot, fairing, incrementing, and cross plotting. The figures used are either photographic reproductions of the screen display as seen by the user or photo enlargements of the 35 millimeter microfilm used in the generation of hardcopy. The user's view of the screen is of white lines on a dark blue background (IBM 2250) or green lines on a dark green background (Vector General DD2).

However, for clarity and reproducibility the photo negatives have been reversed to produce a black on white image.

### Selection of Data

Following entry of certain accounting information, the user is presented with a display listing the data sets currently stored on the data base (fig. 8) from which the appropriate data set is selected by light pen detection (LPD). (S-3A HIGH SPEED was selected for the example to follow.) This action results in the TABLE OF CONTENTS display of figure 9 where TEST TABLE is normally LPDed at this point. The resulting TEST TABLE (fig. 10) consists of a list of the wind tunnel tests contained in the data set selected from figure 8. Limited information about the test such as type, date, or test facility may be included in addition to the test number. LPD of one of the test numbers in figure 10 brings up a RUN SCHEDULE (fig. 11) displaying 20 runs at a time. This display may be rolled up or down by LPD to display desired runs. Runs to be plotted are singled out by LPD of their run number upon which a small arrow appears to the left of the word RUN (fig. 11). In the example problem, runs 6, 17, 33, and 48 from test N277 were required to be displayed on the same plot with certain runs from test N289.

When all the N277 runs had been selected TEST SCHEDULE was LPDed from the bottom menu (fig. 11) to return to the figure 10 display where N289 was detected. After the last run of those desired from N289 (7, 71, and 105) was selected (fig. 12), the word PLOT was LPDed in the lower menu to complete the process of selection of data to be plotted.

### Plotting and Fairing of Basic Data

The PARAMETER TABLE, figure 13, obtained in this example by LPD of PLOT in figure 12, allows the user to select the parameters to be plotted and the grid and size of paper to be used. The grid and paper size default to metric and  $8\frac{1}{2}$  by 11 inches unless overridden by LPD. While not readily apparent in figure 13, the METRIC GRID and A SIZE are brighter than the other options. If INCH GRID is detected, it then becomes the brighter display. The metric grid available for plotting is 16 centimeters horizontally and 20 centimeters vertically, and for B the horizontal size is 26 centimeters. The inch grid is 6 inches horizontally and 8 inches vertically for A size, and for B size the horizontal size is 10 inches. (1 in. = 2.54 cm.)

The parameter for the horizontal axis is selected first by LPD from the list displayed under PARAMETERS. Then the vertical axis parameter is LPDed. As each parameter is selected the center of the screen fills in to give further information, as shown in figure 14. In this example, angle of attack, ALPH, and drag coefficient, CD, have been selected for the horizontal and vertical axes, respectively.

During the parameter selection, the program has examined the range of variation of the data from each of the runs previously selected, compared this with the number of cycles of grid available on the paper selected, and computed a non-bastard scale factor (multiple of 1, 2, or  $4 \times 10^n$  UNITS/CYCLE where n is some positive or negative integer) which will display all the data within the grid boundaries. This process, referred to as AUTO SCALE, is automatic.

Figure 15 is the graphic display of basic data points for the runs selected and is the result of selecting PLOT DATA in the PARAMETER TABLE, figure 14. Due to display limitations and for clarity, fewer grid lines are shown on the graphic display than will be shown on the subsequent hardcopy. Grid lines are shown only for each cycle. Twenty times as many lines will be shown on the hardcopy. Axis labeling appears at every other cycle on the graphic display and some labels are in scientific notation. Hardcopy will be labeled at every cycle in engineering notation. Certain alphabetic letters are used as symbols in the graphic display. The corresponding hardcopy will utilize different symbols.

In addition to the curve symbol identification menu at the upper right and the three boxed menus at the bottom, there are four arrows at the lower right. These may be used to center the plotted points within the grid area. LPD of any of the arrows shifts the display by one half cycle in the direction of the arrow.

LPD of the word FAIRING in the lower center menu of figure 15 brings up the fairing menu shown at the bottom of figure 16. The LINEAR option connects each consecutive point with a straight line. The LEAST SQUARES option generates a least squares polynomial fit up to seventh order. The parameter on the vertical axis is the dependent variable, i.e., the sum of the squares of the vertical distances from the points to the curve are minimized. This is the most suitable and most used option for experimental data. The SPLINE FIT generates a chain cubic fit passing through every point. A different set of coefficients for the cubic are computed between each two points such that the first and second derivatives of adjacent equations have identical values at each data point. This option gives difficulty when two data points are very close, as is the case with "repeat points" which are often taken in each run of a wind tunnel test as a check for hysteresis. NEW METHOD is an option similar to SPLINE but is slightly better behaved.

If LEAST SQUARES is selected, as in this example, a further display permits selection of the degree of the polynomial to be used upon which the display is restored to that of figure 15, except that the word DRAW appears just below the upper right CURVES menu. LPD of DRAW, followed by LPD of any of the symbols under CURVES, results in a display of that curve's fairing. Alternatively, DRAW may be detected twice in succession to fair all of the displayed runs simultaneously (see figure 17).



If the fairings displayed are not visually satisfactory there are several methods for altering them, either individually or collectively. These are:

- o Change the order of the polynomial used for LEAST SQUARES FIT.
- o Change the type of fit.
- o Delete points from consideration in the fairing process.
- o Add points to the data. This is time consuming and seldom used.
- o Move selected points insofar as the curve fairing is concerned while retaining the ability to display the data points in either their original or altered positions.

When a satisfactory set of fairings has been obtained, a HARDCOPY may be LPDed in the lower center menu of figure 17 to produce microfilm resulting in a faired plot of the basic data, figure 18.

#### Incrementation

In ADAIS incrementation is defined as obtaining differences between the dependent values of one or more curves and a base curve at equally spaced intervals of the independent variable. In the example chosen, an S-3A Viking model was tested at several wing spoiler deflection angles in two different wind tunnel tests. In each test a base run was made at zero spoiler deflection. Determination of the incremental drag due to spoiler deflections ranging from 0 to 60 degrees was desired. This is obtained by finding the difference between each of the non-zero spoiler deflection runs and the one from the appropriate test having zero deflection. In ADAIS nomenclature the non-zero spoiler runs are designated NON BASE and the zero spoiler runs BASE runs. Through light pen interaction with a series of displays not shown, the BASE and NON BASE runs are identified, the incremental dependent variable ( $\Delta CD$ ) is computed (NON BASE minus BASE) at equally spaced values of the independent variable (ALPH). Subtraction between two curves occurs at precisely the same value of the independent variable (ALPH) because the mathematical curve fits previously determined permit accurate interpolation. Figure 19 is the result of this incrementation process prior to fairing. Note that the data points representing the subtraction of RUN 7 from RUN 105, designated by the symbol A, are not completely displayed in figure 19. This is due to display buffer limitations which have been corrected since the photo was made. The undisplayed data are available on any hardcopy taken at this point. Note also that the data are not well centered on the plot in figure 19. This was easily corrected by LPD of the upper arrow in the lower right hand corner of the display. Four LPDs of the arrow resulted in figure 20 where the data is well centered.

Figures 21 and 22 compare least squares curve fits of the figure 20 data using fifth and third order, respectively, and figure 23 shows the hardcopy result from the third order fit of figure 22.

## Cross Plotting

To cross plot it is necessary to define a third variable, such as spoiler deflection angle in the example, a different value of which can be associated with each curve to be cross plotted. What this means in terms of figure 22 is that values of 0, 7.5, 15, 30, 45, and 60 degrees were assigned to the curves from bottom to top. Through a series of steps(not shown) involving use of both the light pen and typewriter keyboard the cross plot of figure 24 is produced. Figure 24 illustrates a fifth order least squares polynomial fit and figure 25 shows the same data fitted by third order, the fairing ultimately chosen. Figure 26 is the hardcopy resulting from figure 25. Figure 26 thus represents the incremental drag coefficient versus spoiler deflection for lines of constant angle of attack, ALPH. Each curve can be identified as to its ALPH value by correlating the curve symbol with the symbol key in the upper right corners of figures 25 or 26 where the ALPH value (0, 2, 4, 6, or 8) is shown below the symbol.

## IMPROVEMENT POTENTIAL

### Interfaces to Other Programs

Much of the data extracted from wind tunnel tests by the methods described above are destined to become input data for various performance or stability and control computer programs, some of which are also extant on analytic graphics. To conserve storage and speed computation many of these programs accept data in terms of coefficients for polynomial expressions. Presently in development is the capability to provide coefficients of the least squares polynomial fit equations by 1) visual display on the screen, 2) print output, 3) punch card output, and 4) a created data set accessible from other analytic programs.

The level of effort to process wind tunnel data into a format suitable for loading onto the data base is greatly reduced when there is coordination between the ADAIS programmer and those responsible for running the test prior to the test. Continued publicizing of this fact is expected to improve the interface between the wind tunnel data reduction programs and ADAIS.

### Utilization

Increased utilization of the program is desirable to amortize its development costs at an earlier date. To a large degree ADAIS utilization parallels the frequency of wind tunnel testing at LCC. Timing was such that ADAIS development occurred too late to have a significant impact on the relatively large wind tunnel test programs of the L-1011 Tristar and S-3A Viking. However, it is believed that better documentation of the

program might have improved recent ADAIS utilization. Accordingly, the authors developed a comprehensive 120 page users manual, published in August 1975, and distributed it widely among potential users. Its impact on utilization is still to be assessed.

Lack of advance preparation by the user has resulted in some sessions being less than cost effective. Warnings and admonitions in the above mentioned users manual are expected to alleviate this problem.

As a result of geographic moves of engineering personnel among various engineering buildings it has become necessary for many of the potential users to make a five to ten minute walk to a building containing a graphics terminal. Although the deterrent effect of this inconvenience has not been quantified, efforts continue to be made to locate graphics terminals in buildings where the bulk of the users are based.

Quality control of hardcopy and, to a lesser extent, microfilm quality have been continuing problems. Efforts are being made to justify in-house hardcopy generation capability which should provide more control over quality as well as possible reduction in elapsed time between the graphics session and receipt of hardcopy which is currently running between 24 and 48 hours. Equipment other than the Datagraphics 4060 is also being considered as a microfilm generator.

#### Expansion of Applications

Underway is the expansion of the ADAIS concept into a non-engineering area, namely that of financial forecasting. This application envisions a data base consisting of manpower and materials performance on previous projects which will be readily accessible and manipulatable for rapid response during bid and proposal activity. Plots suitable for direct inclusion in proposals are a desired result.

Use of ADAIS for analysis of pressure data obtained from wind tunnel testing is restricted by the limit of 12 variables per run, whereas 200 to 300 pressure values per test point are not uncommon for some pressure models. Hopefully, a potential user will be able to justify the cost of restructuring the data base to accommodate a variable number of parameters at each test point.

#### CONCLUSIONS

ADAIS is a highly interactive analytic graphics application program which utilizes the full capabilities of the graphics equipment. It has been human engineered to maximize use of the light pen in order to reduce the distractions from frequent shifts of attention to the typewriter or function keyboards.

Productivity using ADAIS of five times that for conventional manual methods of wind tunnel data analysis is routinely achieved. As many as 45 useful plots per hour have been produced by experienced users.

The broad applicability of the ADAIS system is demonstrated by prior use of thermodynamic, basic loads, and flight dynamics data bases as well as by projected development into non-engineering applications.

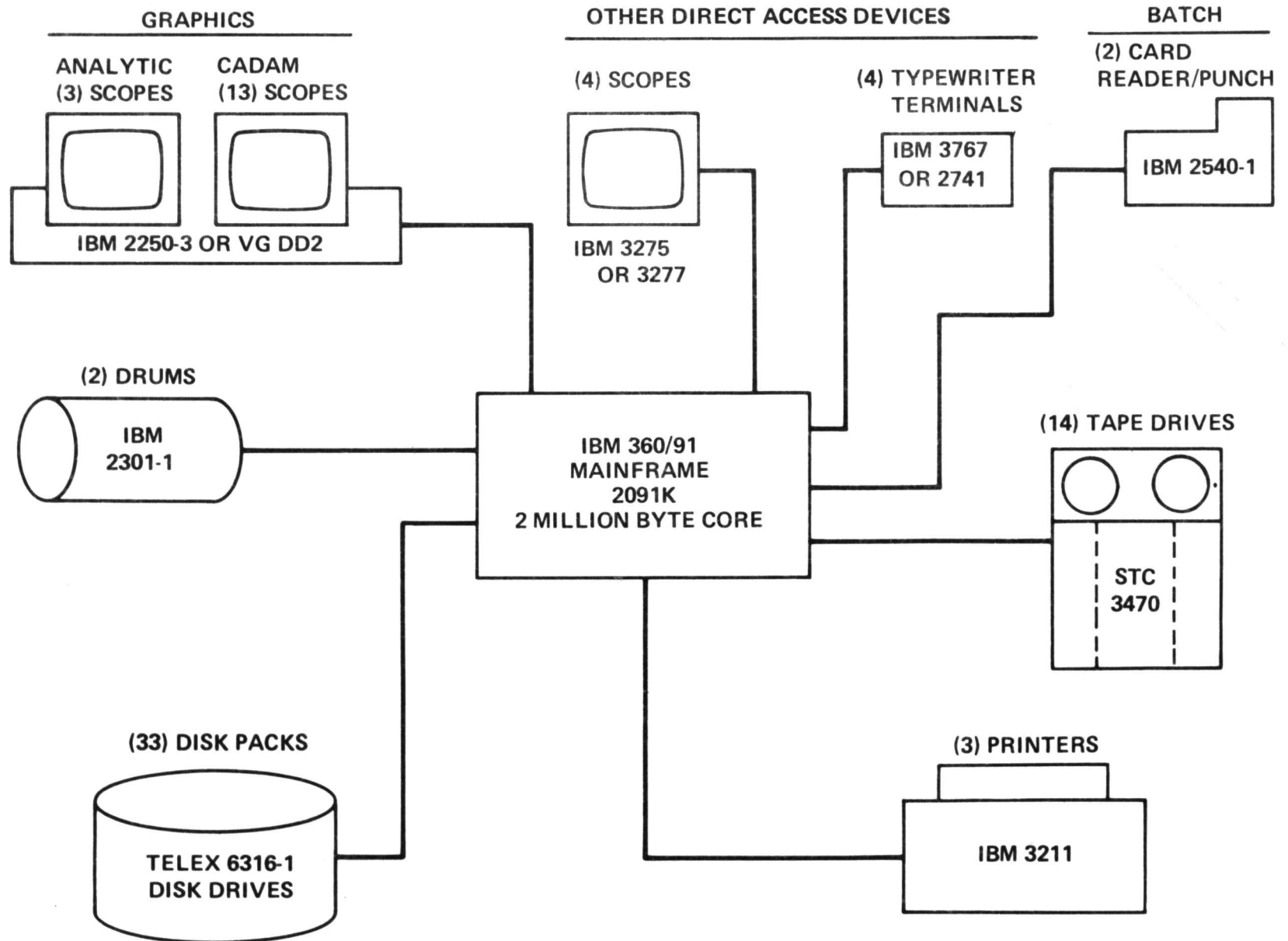


FIGURE 1. SCHEMATIC OF LOCKHEED-CALIFORNIA COMPANY IBM 360/91 INSTALLATION

CPU ACCESS  
HIERARCHY

BYTES

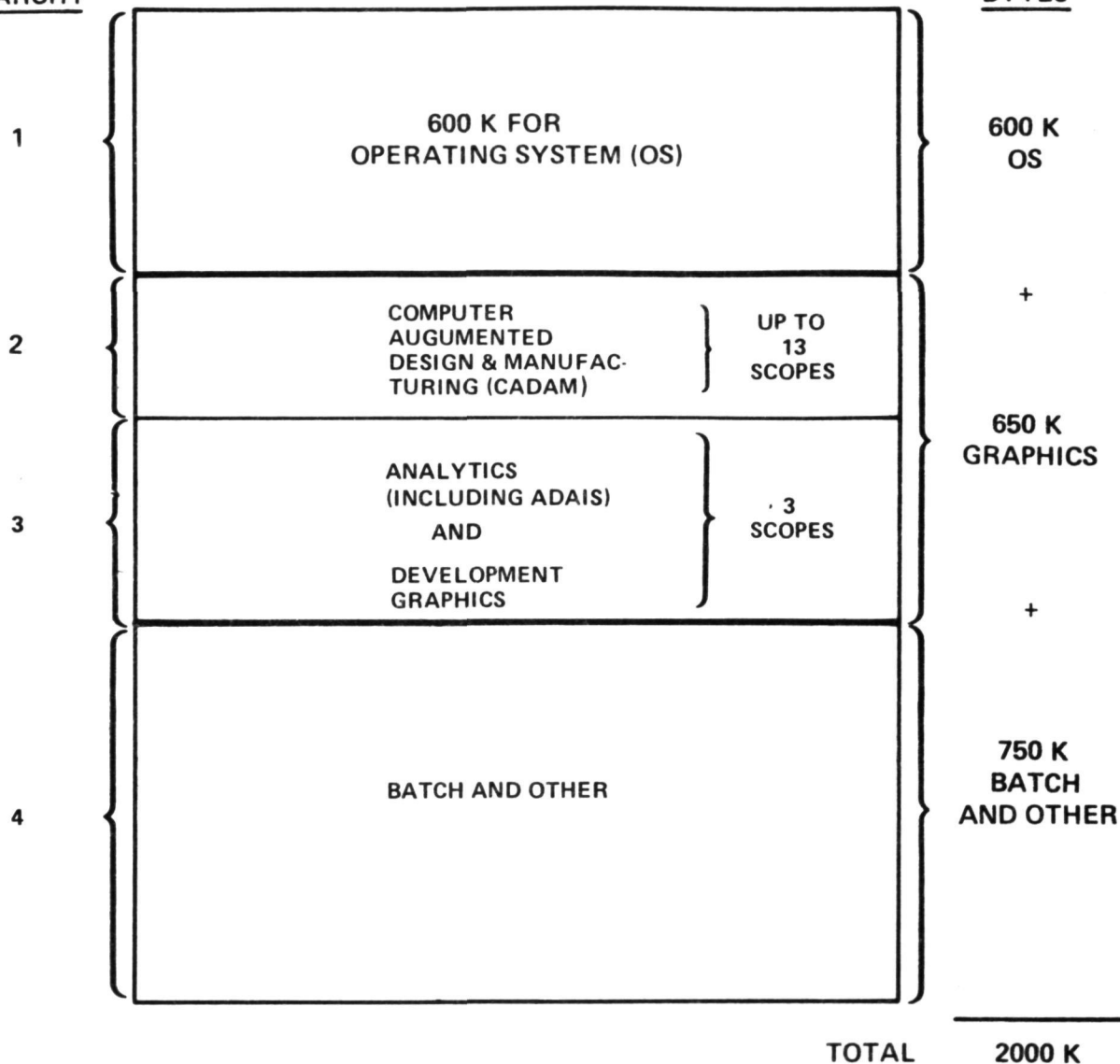


FIGURE 2. TYPICAL IBM 360/91 CORE PARTITIONING - DAY SHIFT

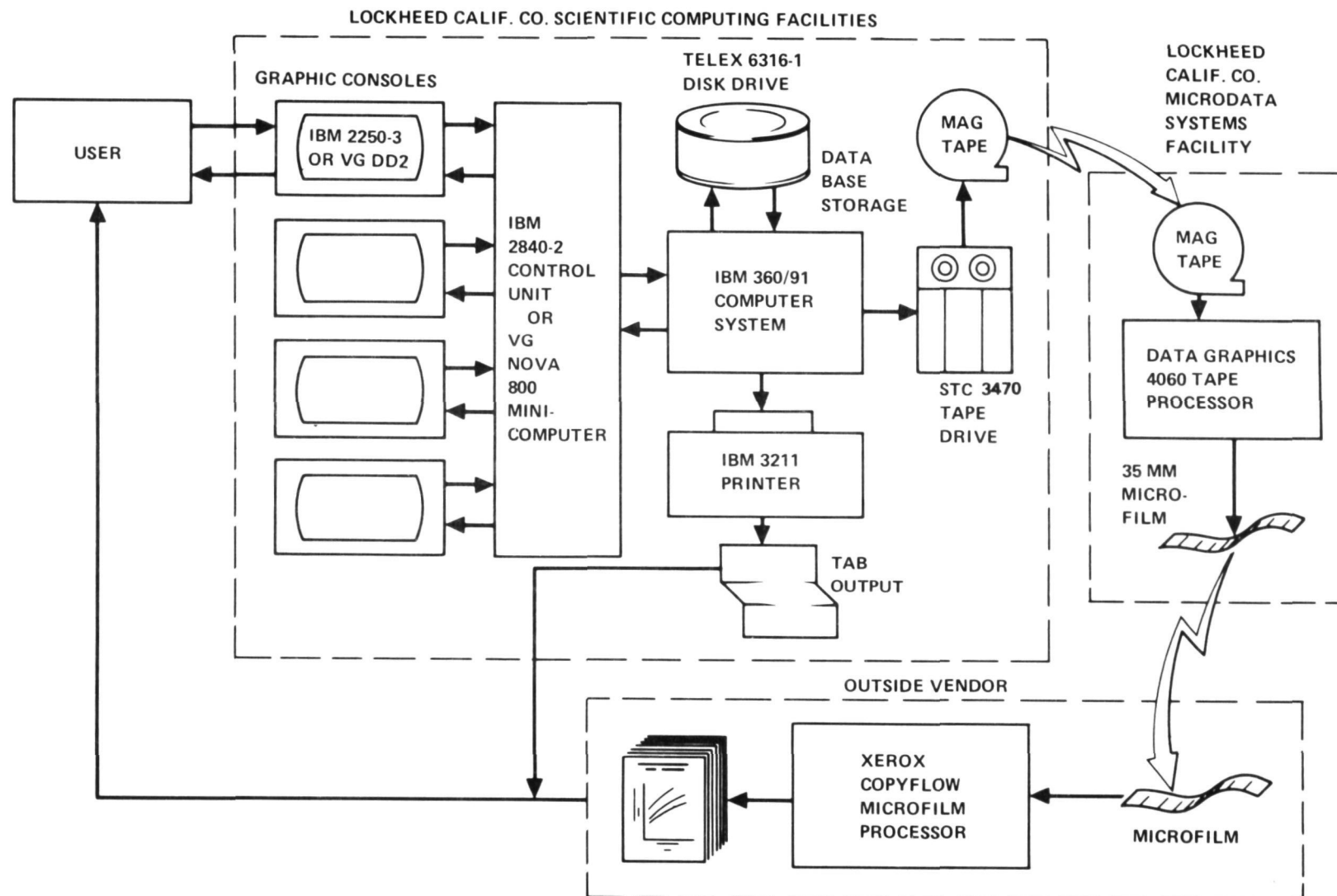


FIGURE 3. ADAIS HARDWARE ENVIRONMENT

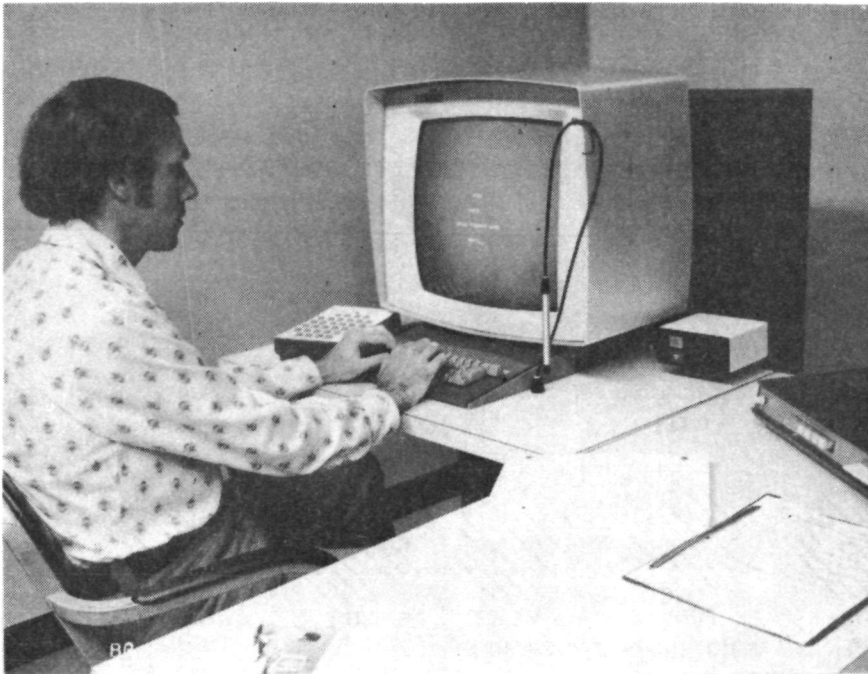


FIGURE 4. IBM 2250-3 DISPLAY CONSOLE

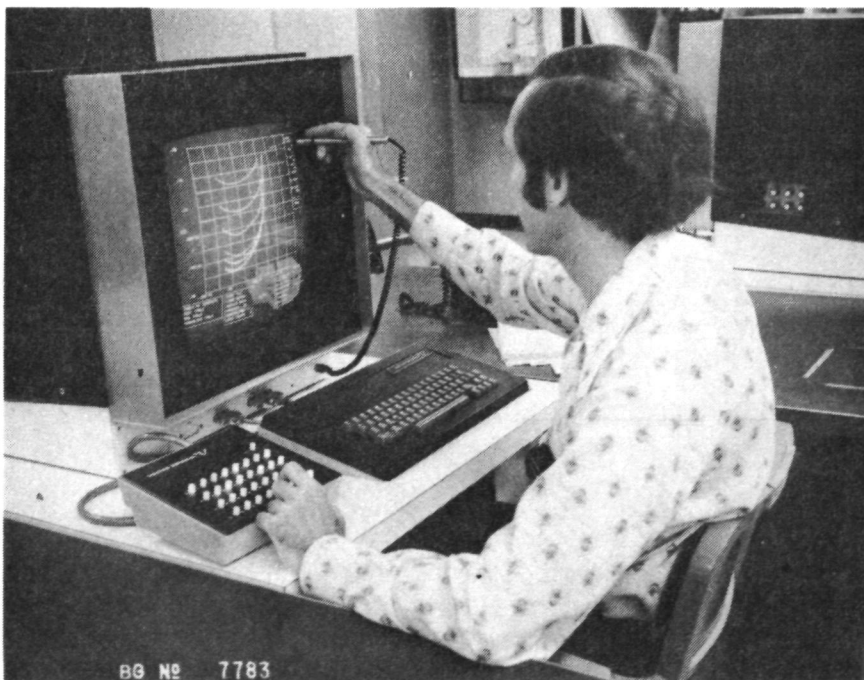


FIGURE 5. VECTOR GENERAL DD2 DISPLAY CONSOLE



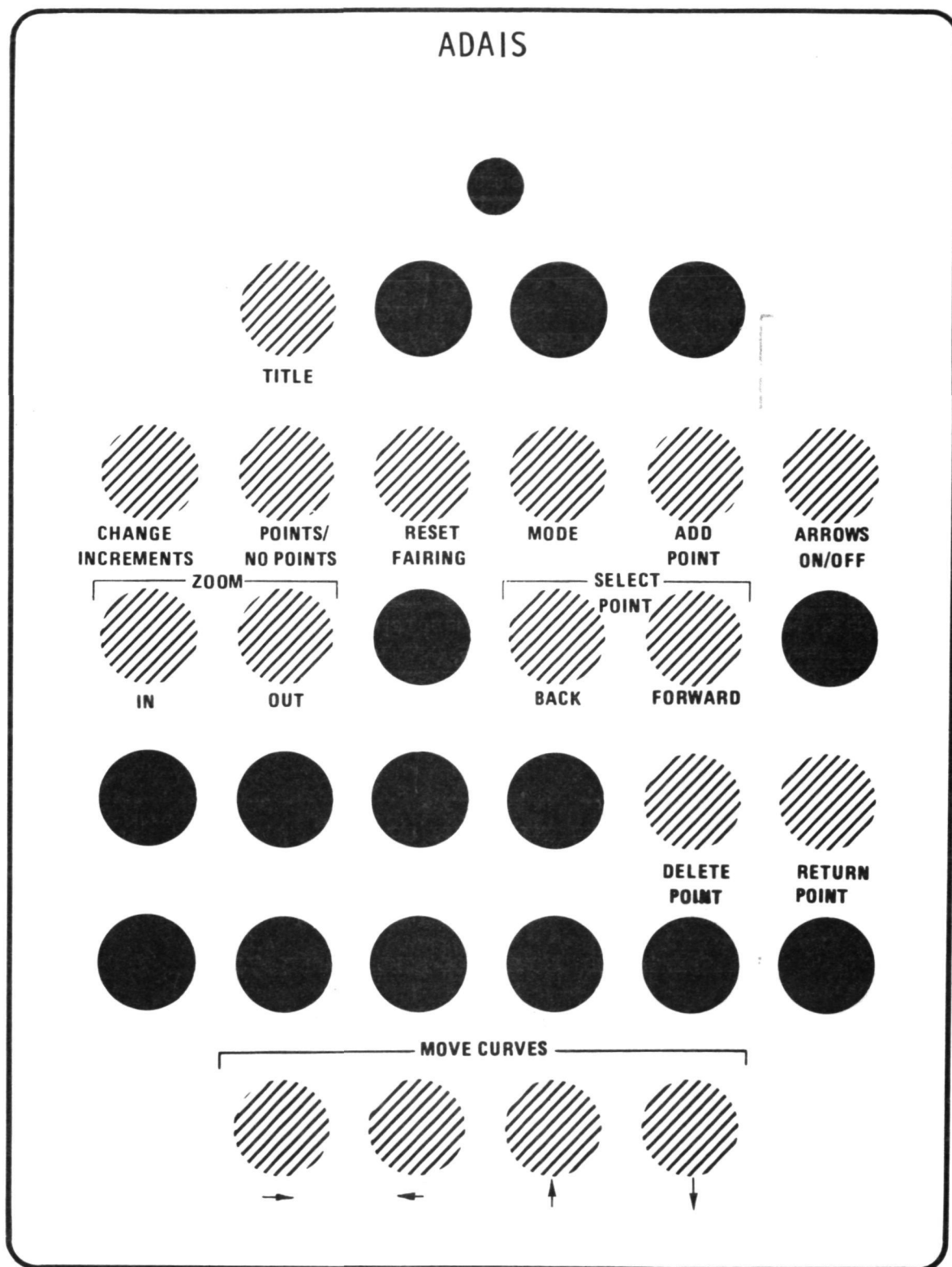


FIGURE 6. IBM ADAIS FUNCTION KEYBOARD OVERLAY TEMPLATE

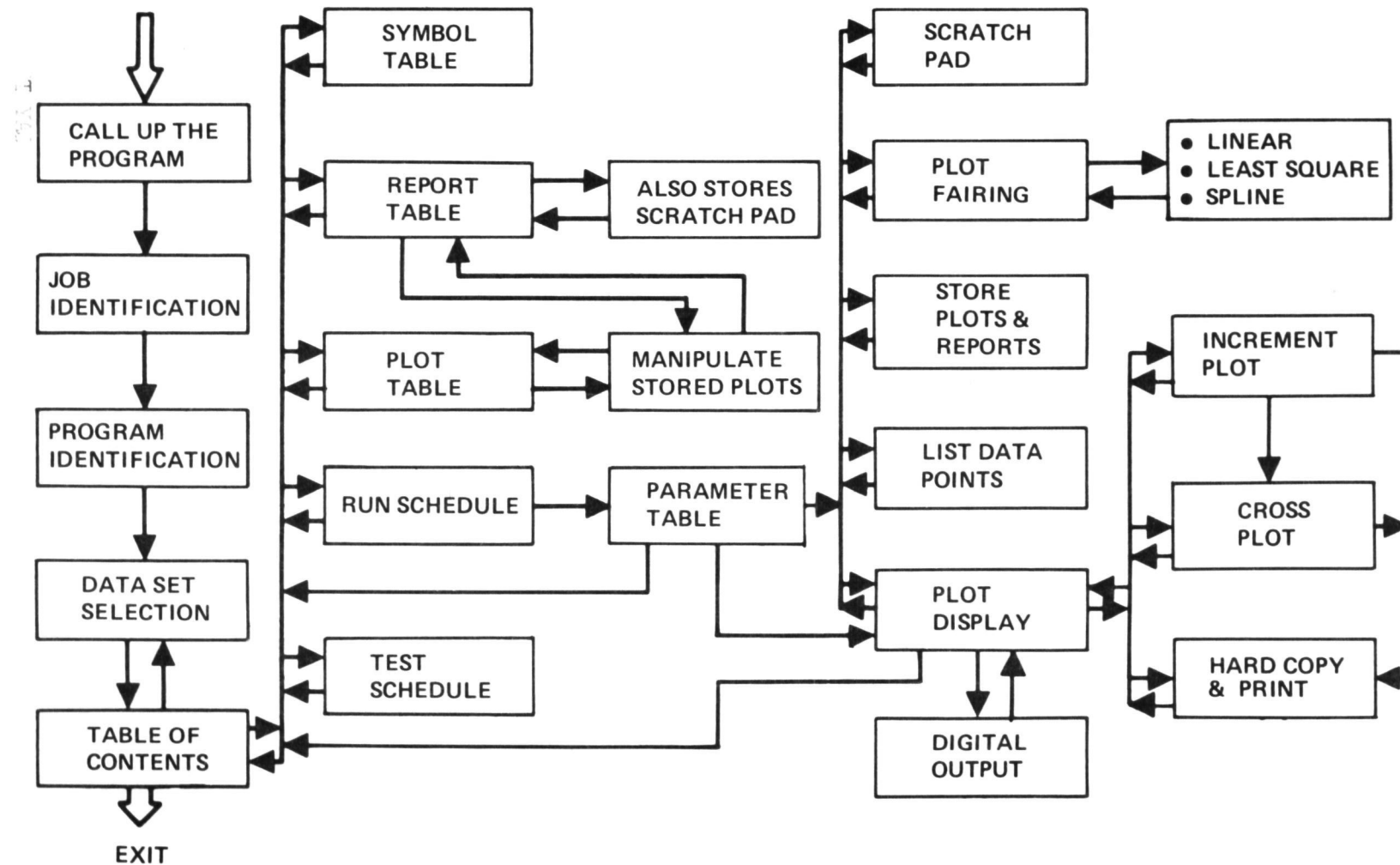


FIGURE 7. ADAIS MAIN LOGIC FLOW

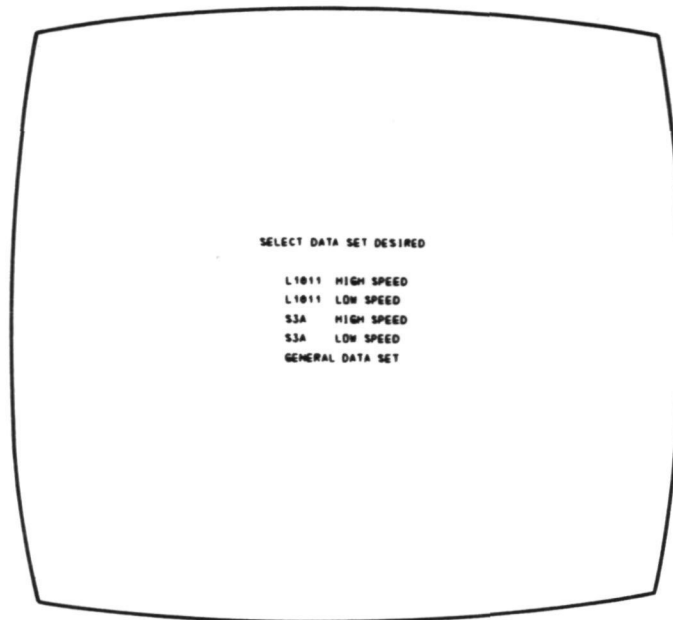


FIGURE 8. DATA TEST SELECTION

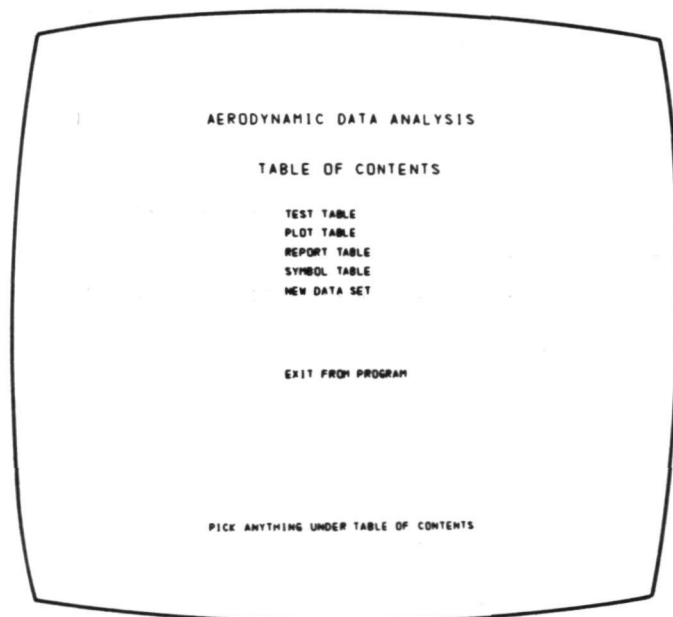


FIGURE 9. TABLE OF CONTENTS

TEST TABLE			
TEST	PLACE	DATE	TYPE
N249	S-3A	HIGH SPEED TEST(CORNELL)	
N254	S-3A	HIGH SPEED TEST(CORNELL)	
N281	S-3A	HIGH SPEED TEST(AMES)	
N261	S-3A	HIGH SPEED TEST(AMES) 12/5/69	
N242	S-3A	HIGH SPEED TEST(AMES) 2/18/70	
N274	S-3A	HIGH SPEED TEST(AMES) 5/12/70	
N284	S-3A	HIGH SPEED TEST(AMES) 7/1/70	
N277	S-3A	HIGH SPEED TEST(AMES) 7/10/70	
N230	S-3A	HIGH SPEED TEST(CORNELL)	
N252	S-3A	HIGH SPEED TEST(CORNELL)	
N243	S-3A	HIGH SPEED TEST(CORNELL)	
N244	S-3A	HIGH SPEED TEST(CORNELL)	
N250	S-3A	HIGH SPEED TEST(CORNELL)	
N278	S-3A	HIGH SPEED TEST	
N229	S-3A	HIGH SPEED TEST (CORNELL)	

TABLE OF CONTENTS

PICK ANY TEST TO DISPLAY TEST RUN SCHEDULE OR TABLE OF CONTENTS TO RETURN

FIGURE 10. TEST TABLE

RUN SCHEDULE OF TEST - N277						
RUN CONFIGURATION MACH						
RUN	21	CONFIG MACH	0.652	SWEEP ALPHA	14	DATA POINTS
RUN	22	CONFIG MACH	0.852	SWEEP ALPHA	14	DATA POINTS
RUN	23	CONFIG MACH	0.801	SWEEP ALPHA	14	DATA POINTS
RUN	24	CONFIG MACH	0.751	SWEEP ALPHA	14	DATA POINTS
RUN	25	CONFIG MACH	0.702	SWEEP ALPHA	14	DATA POINTS
RUN	26	CONFIG MACH	0.652	SWEEP ALPHA	14	DATA POINTS
RUN	27	CONFIG MACH	0.501	SWEEP ALPHA	17	DATA POINTS
RUN	28	CONFIG MACH	0.850	SWEEP ALPHA	14	DATA POINTS
RUN	29	CONFIG MACH	0.802	SWEEP ALPHA	14	DATA POINTS
RUN	30	CONFIG MACH	0.751	SWEEP ALPHA	14	DATA POINTS
RUN	31	CONFIG MACH	0.701	SWEEP ALPHA	14	DATA POINTS
RUN	32	CONFIG MACH	0.651	SWEEP ALPHA	14	DATA POINTS
RUN	33	CONFIG MACH	0.501	SWEEP ALPHA	17	DATA POINTS
RUN	34	CONFIG MACH	0.801	SWEEP ALPHA	14	DATA POINTS
RUN	35	CONFIG MACH	0.851	SWEEP ALPHA	14	DATA POINTS
RUN	36	CONFIG MACH	0.801	SWEEP ALPHA	14	DATA POINTS
RUN	37	CONFIG MACH	0.753	SWEEP ALPHA	14	DATA POINTS
RUN	38	CONFIG MACH	0.700	SWEEP ALPHA	14	DATA POINTS
RUN	39	CONFIG MACH	0.651	SWEEP ALPHA	16	DATA POINTS
RUN	40	CONFIG MACH	0.501	SWEEP ALPHA	18	DATA POINTS

NO. OF CURVES = 3

ROLL UP/ ROLL DOWN/ TABLE OF CONTENTS/ TEST SCHEDULE/ PLOT / PRINT  
PICK RUN TO PLOT OR ITEM FROM MENU

FIGURE 11. RUN SCHEDULE - FIRST TEST

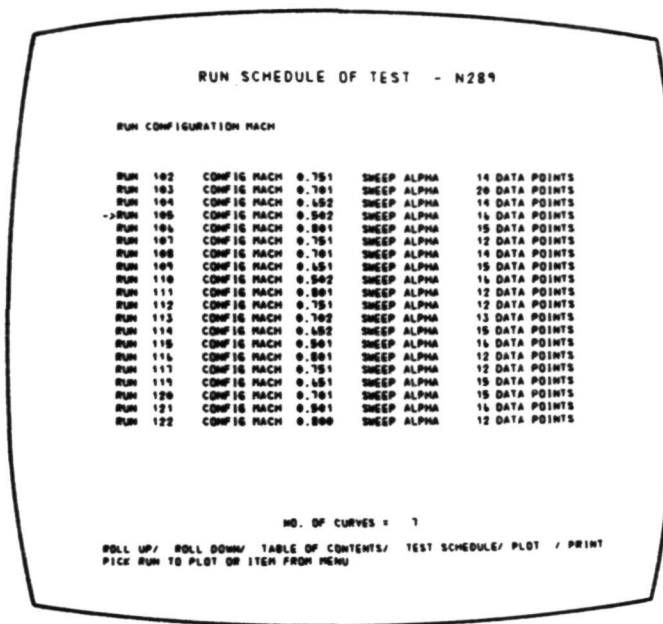


FIGURE 12. LAST RUN OF SECOND TEST DETECTED

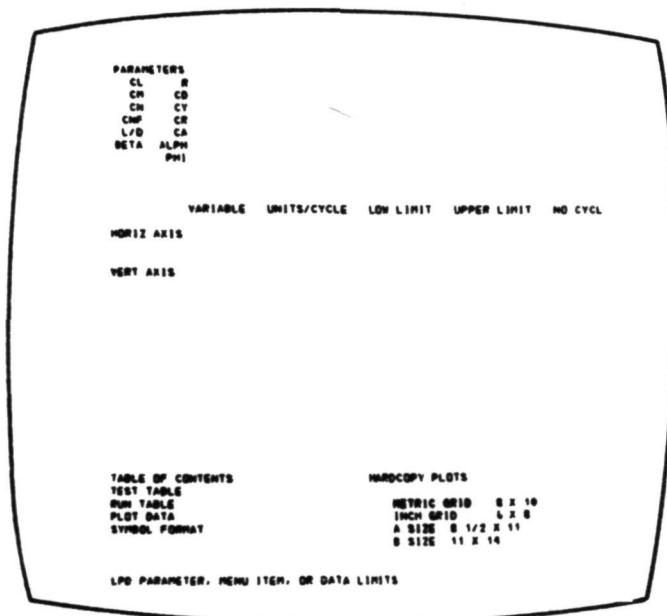


FIGURE 13. PARAMETER TABLE - INITIAL DISPLAY

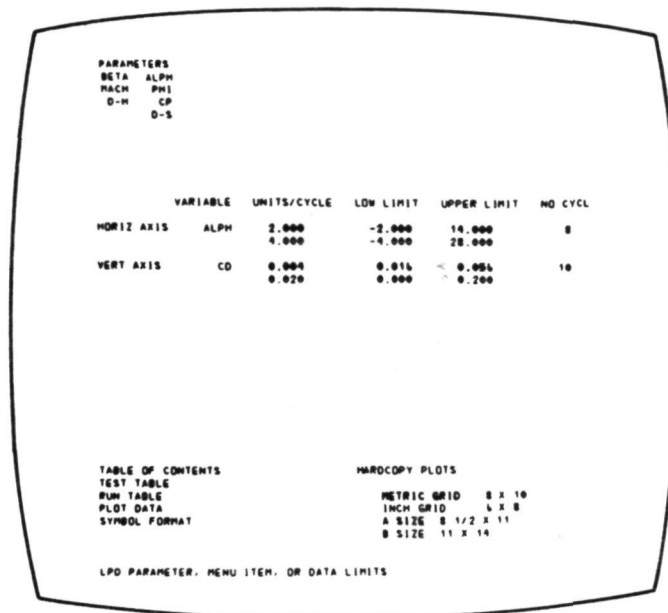


FIGURE 14. PARAMETER TABLE - PARAMETERS DETECTED

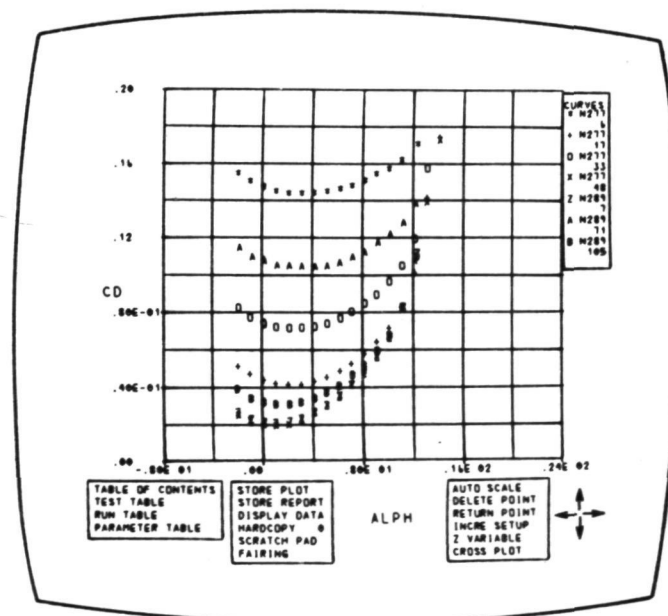


FIGURE 15. BASIC DATA POINT DISPLAY

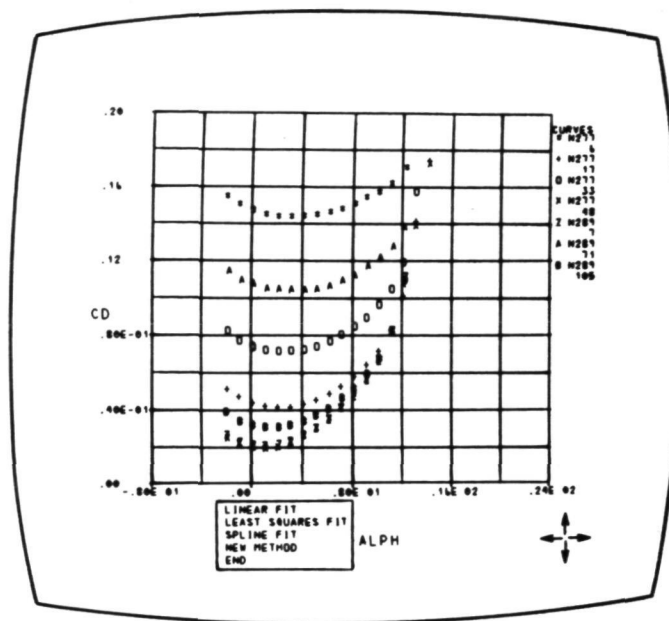


FIGURE 16. FAIRING MENU

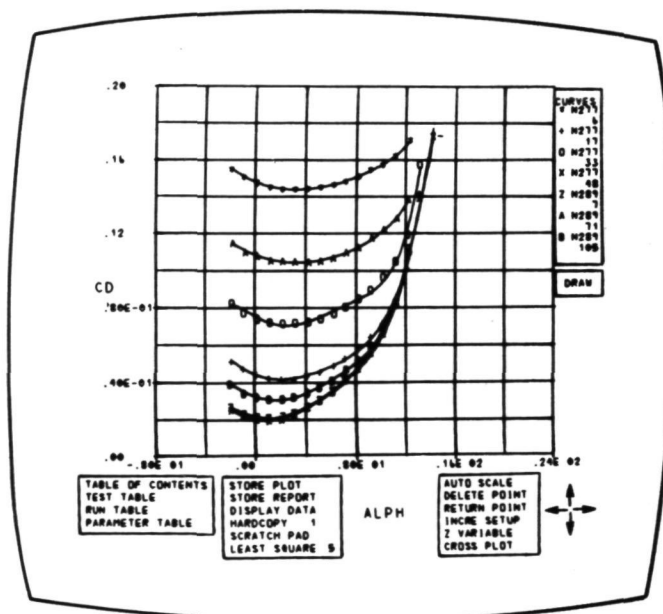


FIGURE 17. FIFTH ORDER FIT OF BASIC DATA

2

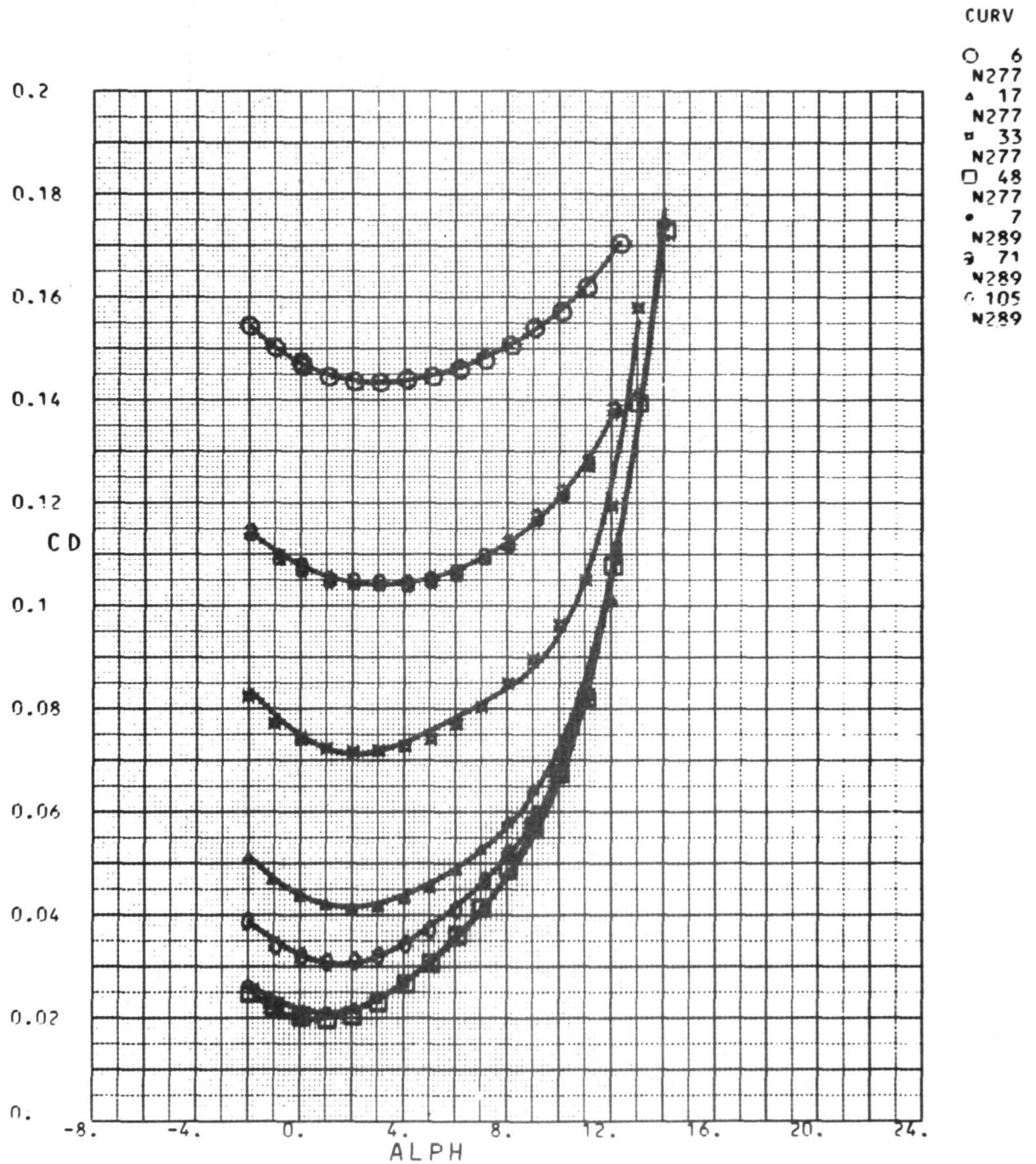


FIGURE 18. HARDCOPY OF FAIRED BASIC DATA



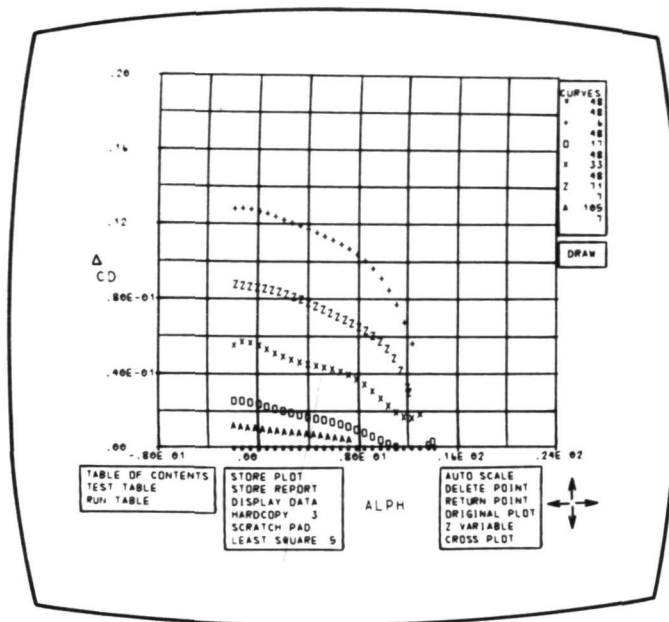


FIGURE 19. INCREMENT PLOT POINT DISPLAY, INITIAL POSITION

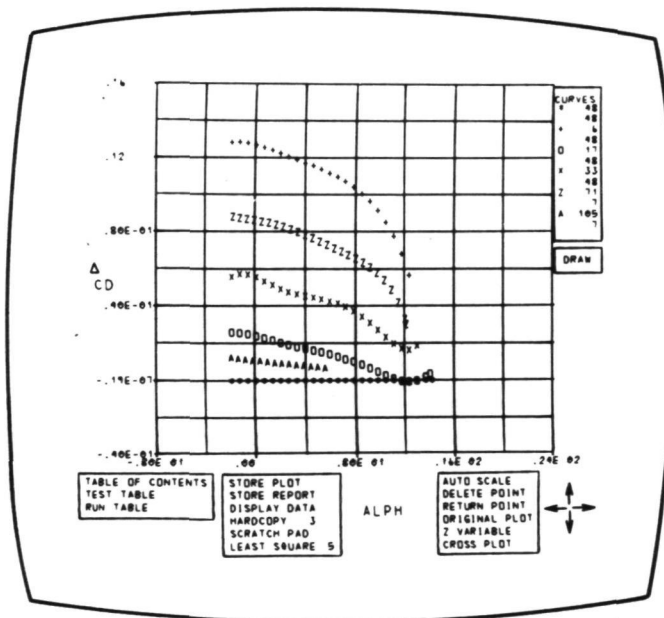


FIGURE 20. INCREMENT PLOT POINT DISPLAY, MOVED POSITION

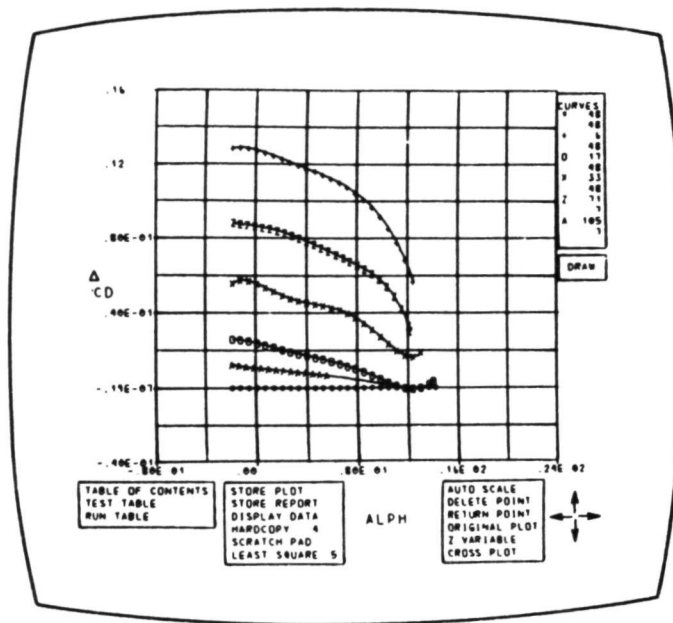


FIGURE 21. INITIAL FAIRING OF INCREMENT PLOT

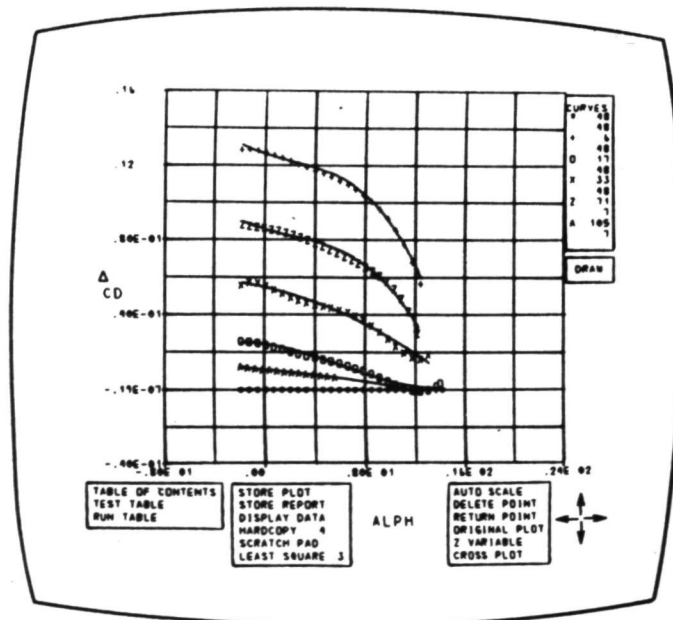


FIGURE 22. FINAL FAIRING OF INCREMENT PLOT

6

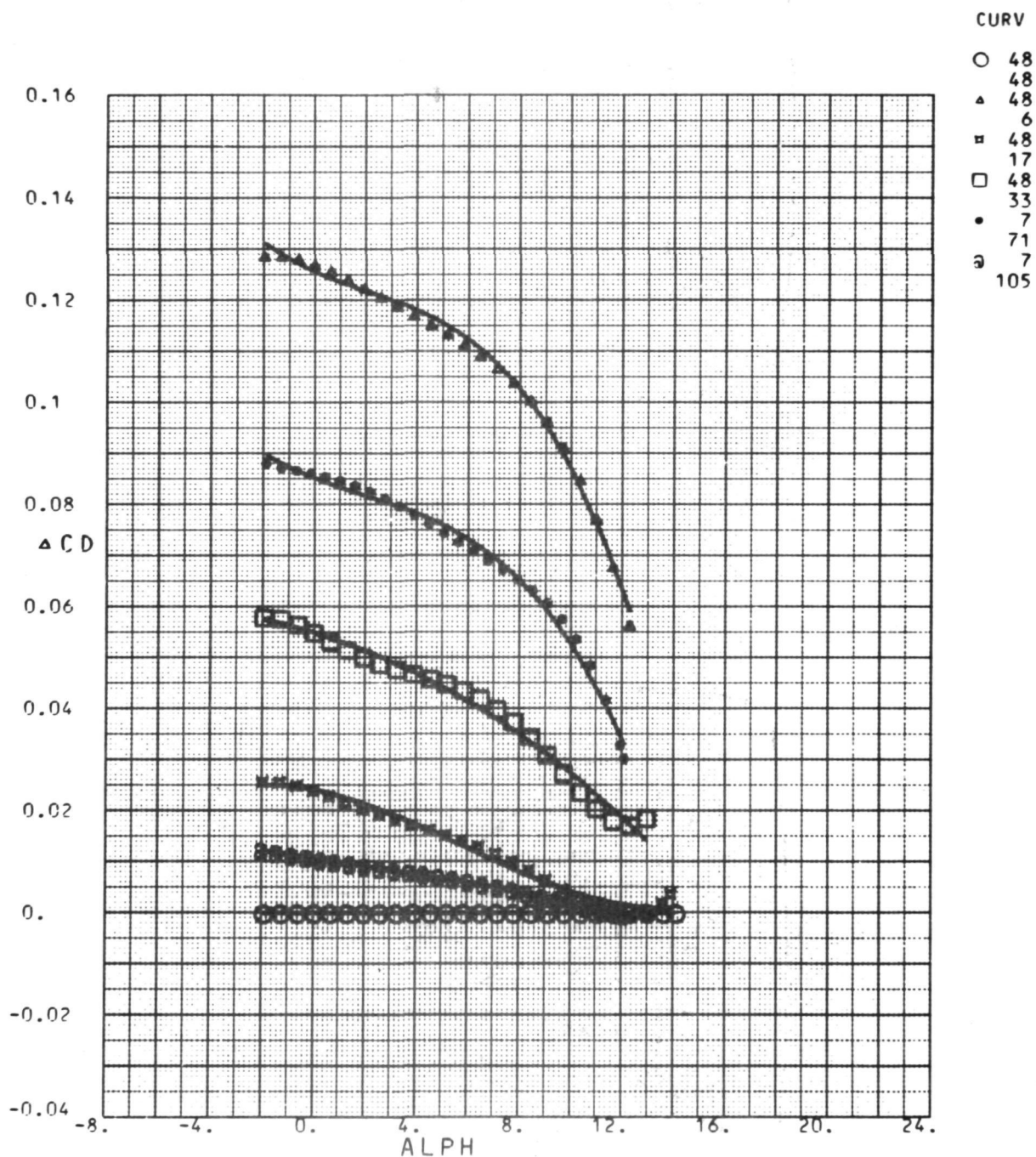


FIGURE 23. HARDCOPY RESULT FOR FIGURE 22

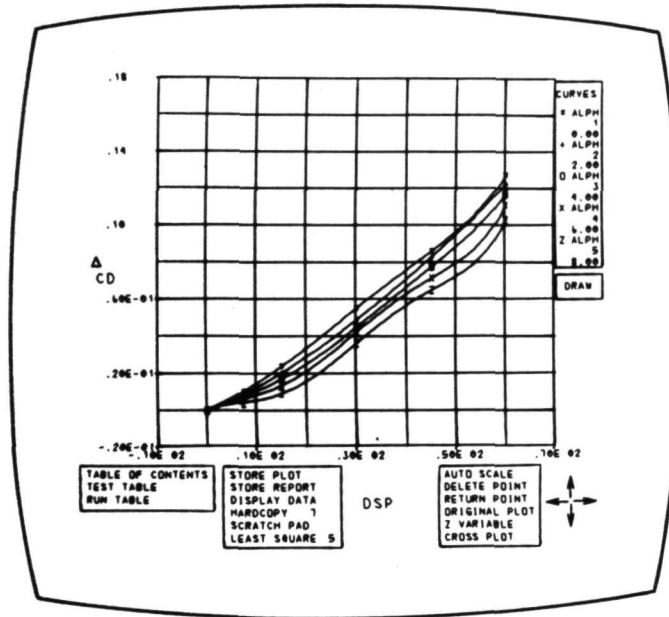


FIGURE 24. FAIRED CROSS PLOT, LEAST SQUARE 5

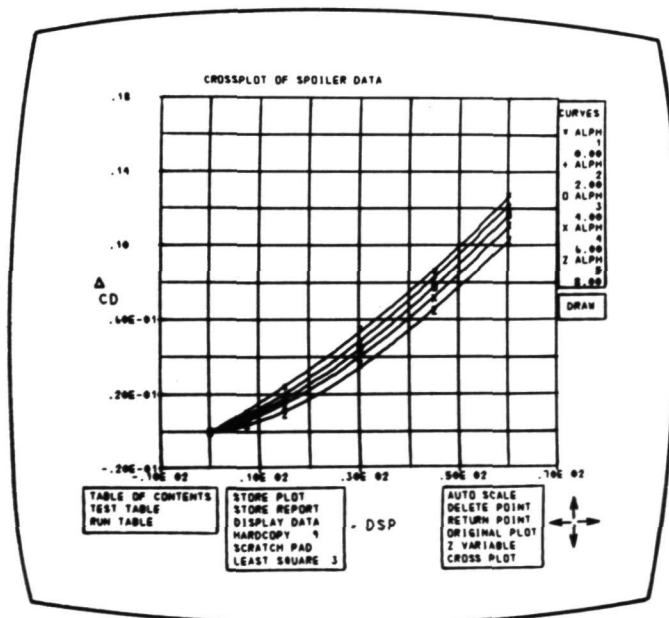


FIGURE 25. FAIRED CROSS PLOT, LEAST SQUARE 3

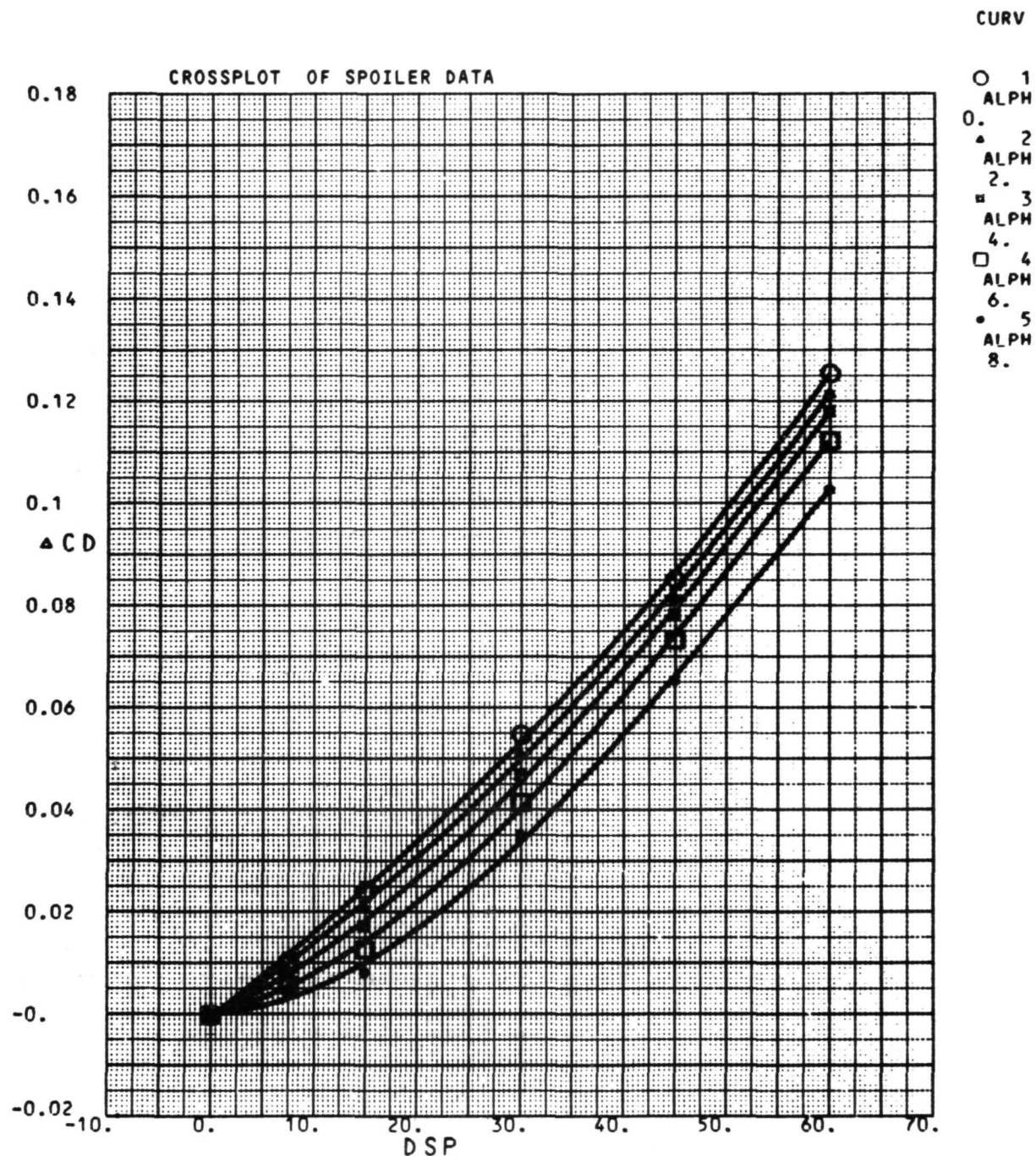


FIGURE 26. HARDCOPY OF FIGURE 25 CROSS PLOT

# APPLICATION OF INTERACTIVE COMPUTER GRAPHICS

## IN WIND-TUNNEL DYNAMIC MODEL TESTING

Robert V. Doggett, Jr.  
NASA Langley Research Center

and

Charles E. Hammond  
U.S. Army Air Mobility R&D Laboratory

### SUMMARY

The computer-controlled data-acquisition system recently installed for use with the Langley Research Center transonic dynamics tunnel is described. This description includes a discussion of the hardware/software features of the system. A subcritical response damping technique, called the combined randomdec/moving-block method, for use in wind-tunnel model flutter testing that has been implemented on the data-acquisition system is described in some detail. Some results using the method are presented and the importance of using interactive graphics in applying the technique in near real time during wind-tunnel test operations is discussed.

### INTRODUCTION

The digital computer is finding more and more applications in wind-tunnel testing as existing tunnels seek to update their data-gathering capabilities and as new wind tunnels are constructed having digital data-acquisition systems. The attractiveness of the digital computer in a data-acquisition role, as in most other roles, is its ability to rapidly acquire and process large amounts of data. This capability provides the wind-tunnel test engineer with the ability to examine reduced data on-line and thus the engineer can make better informed judgments as to whether to move to the next test point or repeat the previous test point. Also, computer control and monitoring of analog devices will free the engineer of these routine, yet time-demanding tasks, and give him more time for data analysis.

Presented in this paper is a description of the recently installed computer-controlled data-acquisition system of the Langley Research Center transonic dynamics tunnel, and a discussion of how the interactive graphics capability of this system is being applied to wind-tunnel flutter model studies.

## LANGLEY TRANSONIC DYNAMICS WIND TUNNEL

Before describing the computer-controlled data-acquisition system for the Langley transonic dynamics tunnel, it will serve a useful purpose to briefly describe this wind tunnel and the purposes for which it is used. This description will provide a background for placing the data-acquisition system in the proper perspective. The transonic dynamics tunnel is located at the NASA Langley Research Center. An aerial view of the facility is presented in figure 1 along with two drawings which show the general arrangement of the wind tunnel. The tunnel is relatively large with a 4.88-m (16-ft) square test section. The tunnel is capable of continuous operation, and both tunnel speed (Mach number) and pressure (altitude) are continuously controllable. The building shown in the foreground of the photograph provides office space and shop facilities for tunnel personnel. Most of the data-acquisition system equipment is located on the third floor of this building. Some equipment is located in the control room where the test engineer is usually stationed during wind-tunnel operations, and some equipment is located in a setup/calibration laboratory located on the first floor so that the system can be used in preparing models for subsequent wind-tunnel tests.

The transonic dynamics tunnel is almost totally dedicated to dynamic model testing as opposed to the measurement of static forces and pressures that is done in most wind tunnels. A large number of tests use dynamically scaled aeroelastic models of advanced aerospace systems, including both aircraft and launch vehicles. Some tests are used to provide data on specific configurations while others are of a basic research nature. For the most part, the tunnel tests fall into one of seven categories. A listing of the categories is presented in figure 2. Also shown in the figure is the type of model measurements required, typical number of data channels required, and frequency range of interest. Prior to the installation of the data-acquisition system, each type of test essentially required a unique set of instrumentation. Since a flutter test might be followed by a gust test which, in turn, might be followed by a helicopter test and so forth, a new instrumentation system including signal conditioners, amplifiers, and display and recording equipment had to be set up for each test. This not only resulted in a lengthy setup time for each test, but also meant that tunnel personnel had to be familiar with and expert in the use of many different types of instrumentation. Also, during the tests, considerable effort was necessary to manually adjust such things as amplifier gains to keep data signals within allowable limits. During each test voluminous amounts of information are obtained. Since practically all data reduction was done after the tests were completed, the test engineer had very little reduced data available on-line to guide the conducting of the test.



## COMPUTER-CONTROLLED DATA-ACQUISITION SYSTEM

### Basic Design Requirements

The purpose of the computer-controlled data-acquisition system is to improve the efficiency of wind-tunnel model testing. Here improved efficiency does not mean that more wind-tunnel tests will be conducted in a given period of time, although the use of the system will undoubtedly expedite some tests, but rather it means that the quality and quantity of the data obtained will be improved. Basically, the system was designed to perform four rather general functions in a manner to be specified and controlled by the test engineer. These functions are (1) automatically adjust amplifier gain and offset to make maximum use of amplifier dynamic range, (2) acquire and format tunnel and model test data and record the information on magnetic tapes, (3) process and display in real-time wind-tunnel parameters, and (4) process and display in real time, or near real time, selected model response data. In performing these four functions it was required that the system be very flexible, or versatile, so that it could be used with all of the seven categories of tests previously mentioned as well as meet the needs of future, different test requirements. Another requirement was that the system would be convenient to use, including being relatively easy to learn how to use. Also, advantage would be taken of man-machine interaction where man's decision-making capabilities are desirable, and yet take advantage of the system hardware/software capabilities to free the test engineer from as many routine tasks as possible.

### System Description

In this section the basic components and features of the computer-controlled data-acquisition system will be described. However, no attempt will be made to completely describe the system. For convenience, the data system may be thought of as being composed of five major subsystems which are the Analog Front End Subsystem, Analog-to-Digital Subsystem, Computer Subsystem, Control and Display Subsystem, and Digital Input/Output Subsystem. The interconnection of these five subsystems is illustrated schematically in figure 3. The operation of the system is under the control of the test engineer through the use of appropriate computer software. Provision has been provided in the system for certain hardware status information to be made available to the computer. This information includes such items as analog amplifier gain settings. Also, when suitably programed, the computer can be used to control most of the hardware equipment such as adjusting amplifier gain setting. Data flow paths are shown as solid lines in the figure, command or control signal paths by dashed lines, and status information paths by long and short dashed lines. Analog signals from the wind-tunnel model enter the data system through the Analog Front End Subsystem, which contains signal conditioners, amplifiers, tape recorders, and other analog equipment. In this subsystem, the data are recorded on analog tape for later data reduction (i.e., between wind-tunnel test runs). Selected data are passed from the Analog Front End



Subsystem to the Analog-to-Digital Subsystem where the analog data are converted to digital form and passed to the Computer Subsystem for real-time, or near-real-time, processing. The processed data may be recorded on digital tape and/or displayed for the test engineer's examination by the Control and Display Subsystem; for example, a plot on the graphic display unit screen. Digital data, such as shaft encoder outputs, are input to the computer through the Digital Input/Output Subsystem. The test engineer exercises control of the data system by issuing commands through equipment in the Control and Display Subsystem; for example, typing in a message on a typewriter keyboard.

Analog Front End Subsystem. The Analog Front End Subsystem accepts and processes data signals from transducers located on models mounted in either the wind-tunnel test section or in a model setup/calibration laboratory. This dual input feature allows the data-acquisition system to be used during wind-tunnel testing and in preparing models for subsequent testing. This subsystem provides for transducer excitation, signal conditioning, filtering, signal monitoring, and signal recording. The class of components of this system are similar to those found in most analog systems including amplifiers, filters, tape recorders, and so forth. However, this subsystem has many special characteristics to allow the system hardware components to be monitored and controlled by the digital computer. A simplified block diagram of the subsystem is presented in figure 4, and some of the subsystem components may be seen in the photograph in figure 5. The subsystem, as provided, is a complete 60-channel system. Fifty of these channels are called DC channels and have a frequency-response capability from zero to 20 kHz. These channels are for use with strain-gage-type transducers. Ten of the channels, called AC channels, have a frequency-response capability from 1 to 20 kHz. These channels are for use with piezoelectric transducers. Signal conditioning and excitation are provided for each DC channel. Separate conditioning equipment is provided for the test section and calibration laboratory inputs so that models can be set up in the calibration laboratory while another model is undergoing wind-tunnel testing. Each amplifier is equipped with a computer controllable input source selection switch to select any one of three input sources: test section, calibration laboratory, and calibration bus. The computer can also be used to set amplifier gain, and, for the DC channels, offset control is provided to balance out the zero frequency component of dynamic signals to provide maximum use of the amplifier dynamic range. Each amplifier has dual output channels, one unfiltered and one low pass filtered. The unfiltered output is connected to monitor oscilloscopes and to connector panels for use with other monitor equipment if desired. The filtered data are output to the Analog-to-Digital Subsystem (ADS). Two analog tape recorders are available for recording data, a low band recorder (LB) and an intermediate band recorder (IB). The LB tape unit is primarily for use with low-frequency dynamic data, and the 60 input data channels may be frequency multiplexed in groups of 5 onto 12 of the 14 LB tape recorder channels. The test engineer selects the grouping of the various input channels into the groups of five. The IB tape unit is for use with higher frequency data, and 12 channels of data are recorded simultaneously. A time-multiplexing capability is provided so that different groups of 12 data channels can be selected by the test engineer for sequential

recording. Analog tape recorded data can be played back and input to the computer through the Analog-to-Digital Subsystem (ADS). Additional equipment in this subsystem includes time code generator, time code reader, AC calibration source, and DC calibration source.

Most of the hardware devices in this subsystem can be operated under computer control.

Digital Input/Output Subsystem. The Digital Input/Output Subsystem provides a means of inputting digital information into the computer and a means for the computer to monitor and/or control some of the data system hardware components. This subsystem consists of two major components, the digital multiplexer (DM) and the control signal distributor (CSD). The operation of the DM is controlled by the computer. Most of the digital data input to the computer through the DM is information concerning the status of the data system hardware such as various switch position settings that the computer is required to log, for example, amplifier gain settings. The DM also accepts inputs from the time code generator, shaft encoders, and some spare channels are provided. Although the CSD may be operated manually, it is normally operated under computer control. The CSD receives command words from the computer, decodes these words, and routes command signals to the data system hardware components. For example, command signals from CSD can be used to start and stop the analog tape recorders and adjust the amplifier gains.

Analog-to-Digital Subsystem. The Analog-to-Digital Subsystem is used to convert analog data signal outputs from either the data system amplifiers or playback data from the two analog tape recorders to digital form for computer processing. Some spare input channels are provided to accept data from other sources. The maximum conversion rate is 50 000 samples per second. All incoming data channels are not required to be sampled at the same rate.

Computer Subsystem. The main component of the Computer Subsystem is a XEROX Sigma-5 computer with a 48 000-word memory. Computer peripheral equipment includes a random access storage disk, card reader, card punch, line printer, keyboard printer, digital plotter, and three digital tape units, two nine-track units and one seven-track unit.

Control and Display Subsystem. The purpose of the Control and Display Subsystem is to provide a means of displaying, in real time, information that has been processed by the computer for the test engineer's examination while the wind-tunnel test is in progress, and to provide a means for the engineer to exercise his control over the data system by issuing instructions to the computer. The main components of this subsystem are the graphic display unit (GDU) which includes a keyboard and light gun, two typewriter units, numeric displays, and constants switches. The GDU may be seen in the left foreground of the photograph presented in figure 6 which shows an interior view of the wind-tunnel control room. A bank of numeric displays may be seen in the center of the photograph. Most of the communications between the test engineer and the computer are made by using the GDU. Information from the computer is displayed on the GDU screen in

the form of messages or data plots. The engineer issues his commands to the system by using the GDU keyboard. Messages can also be sent and received by using either typewriter. The numeric displays can be used to access particular computer memory locations to display stored data. The constants switches provide a means for the test engineer to change the contents of particular computer memory locations.

System Control Software. The automated data-acquisition system is controlled and operated by the test engineer through the use of a computer software package called Operating Measurements Program (OMP) which operates in conjunction with the standard Sigma-5 computer monitor system. The OMP program may be thought of as a software supervisor which allows the test engineer to exercise control of the system to acquire, display, and analyze wind-tunnel data by using his own software. As a part of the OMP package, a set of user callable subroutines (UCSUBS) have been provided. The basic computer FORTRAN library subroutines are also available for use by the engineer, and this library may be generally thought of as an extension of the UCSUBS. At the time of system delivery, there were 103 of these UCSUBS available. A large number of these subroutines are related to controlling the system hardware; for example, starting a tape recorder. Others are in the analysis category such as generating a Fourier transform. To operate the data system within the OMP framework, the test engineer is required to write a series of FORTRAN language subroutines called first-level user programs, or FLUPS, which access the UCSUBS, basic library subroutines, and/or the input/output services provided within OMP. The FLUPS may be as simple or as complex as necessary to meet the user's requirements. A simple FLUP might be one that starts an analog tape recorder, records data for 10 seconds, and then stops the analog tape recorder. A more complex FLUP might be one that records data on analog tape, rewinds the tape, plays back the tape and digitizes the data, passes the digitized data to the computer, calculates a power spectral density, and displays the result on the GDU. In most instances, the test engineer causes his FLUP to be executed by typing in the FLUP name on one of the three keyboards, typewriter or GDU. Since there are some requirements which are necessary for all wind-tunnel tests, some standard services are built into OMP and it is not necessary that a FLUP be prepared. The determination of wind-tunnel parameters such as Mach number, velocity, density, and so forth, is an example of a built-in service. The OMP program also can be used to assist the engineer in configuring the system hardware during test setup. This is done by the engineer inputting to the computer the desired setting of various hardware device switches. The computer compares the desired setting with the actual setting and advises the engineer of any improperly configured equipment.

#### FLUTTER TESTING WITH INTERACTIVE GRAPHICS

Flutter is a dynamic instability produced by the coupling of structural inertia and stiffness forces with aerodynamic forces to produce a self-excited, usually increasing in amplitude, sinusoidal oscillation that can lead to catastrophic structural failure. Consequently, flutter must

be taken into account in the design of new aircraft to insure that flutter cannot occur within the normal aircraft operating envelope. Although there are many analytical techniques available for flutter analysis, these techniques are unreliable at transonic speeds where flutter is usually most critical. Therefore, for high-speed aircraft, flutter model studies are relied upon heavily to demonstrate that the aircraft has acceptable flutter characteristics. Important uses of the transonic dynamics tunnel are to perform such flutter clearance studies of new aircraft designs as well as to conduct basic phenomenological flutter research studies. Traditionally, flutter model test procedures have been to treat flutter as an event that either occurs or does not occur. The models are actually taken to the flutter condition, and by varying tunnel parameters (Mach number and dynamic pressure), sufficient flutter points are obtained to define the flutter boundary. A typical flutter boundary in terms of the variation of dynamic pressure with Mach number is shown on the left in figure 7. The variation in wind-tunnel conditions in reaching a typical flutter point is illustrated by the dashed line in the figure. Since flutter is not only hazardous to aircraft but can also cause model structural damage, it would be desirable to obtain the flutter boundary without actually experiencing flutter. This is particularly true since flutter models of specific aircraft configurations are rather expensive, often costing in excess of a hundred thousand dollars. One way to determine the flutter condition without actually experiencing flutter is to make subcritical damping measurements of the critical flutter mode at various flight conditions and extrapolate to the flutter condition since the damping becomes zero at flutter. For example, referring to figure 7, if the flutter boundary were approached along the dashed line shown on the left and damping measurements made at each of the conditions marked A, B, C, D, and E by analyzing model response data while the tunnel conditions are held constant, then a damping variation as indicated on the right in the figure can be obtained. The flutter condition can then be obtained by extrapolating the damping curve to zero as indicated in the figure. By repeating this process, the entire flutter boundary can be determined. Unfortunately, the obtaining of damping from model response data is not an easy task, and subcritical damping techniques have not been routinely used in the past. A large part of the difficulty has been associated with the inability to reduce, analyze, and display model damping data in near real time so that the damping can be continuously monitored during the approach to the flutter boundary. The installation of the data-acquisition system has now made it practical to apply subcritical damping methods to flutter tests in the transonic dynamics tunnel. One promising technique which is currently under development depends heavily on the use of interactive graphics. This technique will be described in the following discussion.

### Subcritical Damping Technique

The subcritical damping technique that has been implemented using the data-acquisition system for use in flutter model testing is a combination of two methods, either of which can be used to determine damping under certain conditions. The first method, called randomdec, is applicable to determining the damping of a system excited by a random force. The second

method, called moving block, requires the model to be excited transiently in order to determine the damping. (Transient response as used here means the free decay following the removal of a discrete excitation force.)

Randomdec method. The randomdec method provides a means of obtaining information about an elastic system by examining the system response to random excitation. A description of the method is presented in reference 1 and only the highlights of the technique will be described here. The randomdec method is illustrated schematically in figure 8. The system response is assumed to be composed of three components: the responses to a step input, to an impulse, and to a stationary random force. The system response to a step force is obtained by an ensemble average of a number of response time segments, since the response to random force and to an impulse average to zero. The averaging process for each time sample is started when the response signal reaches a predetermined level, indicated by  $Y_0$  in the figure. The output of the randomdec analysis has the appearance of a damped sine wave and is referred to as the randomdec signature. From this signature, the system damping and frequency can be obtained. Although the randomdec method has many advantages, the most notable being that it is not necessary to provide external excitation as required by many subcritical damping techniques, its application is not without problems. One difficulty occurs when the system under study has two or more resonances that are relatively close together. This problem can be partially overcome by filtering the response data, but this is not always practical since it implies a certain knowledge of the flutter characteristics beforehand. When the randomdec signature is composed of the response of two or more resonances, beats appear in the signature and make it difficult to obtain the characteristics of any single mode. Some work has been done in applying curve-fitting methods to analytically separate the signature into frequency components. Although this work shows promise, current indications are that effective curve-fitting methods are too slow for near real-time applications such as required for flutter testing.

Moving-block technique. The moving-block technique is a means for obtaining the damping of an elastic system from its transient response. A detailed description of this technique is presented in reference 2. Early applications of the method in rotary-wing stability testing are described in references 3 and 4. The technique is illustrated schematically in figure 9 and may be described briefly as follows: A response signal from the model which has been excited transiently in the mode of interest is first obtained. A Fourier transform of this signal trace is then made to determine the frequency content of the signal, and the analyst selects the model frequency for which the damping calculations are to be made. After optimizing the operator-input frequency to obtain the peak value of the transform nearest the input frequency, a discrete Fourier transform is made with a block of data which is a fraction of the originally collected data block, and the natural log of the amplitude of the discrete transform is taken. A plot of this value against time is constructed by moving the block down the signal trace incrementally one data sample (hence, the name moving block) and repeating the discrete transform, natural log operation until the block reaches the end of the original data sample. If the signal



is a pure damped sine wave, the plot generated will be a curve which oscillates about a straight line. The slope of the straight line determines the damping.

Combined Randomdec/Moving-block technique. Although the moving-block method is suitable for determining damping from transient response time histories, it is not, in general, applicable to flutter model testing since provision for external excitation is not usually provided for in these models. However, there is no difficulty in obtaining a randomdec signature for such models since residual wind-tunnel turbulence provides the required random excitation. Since the randomdec signature may be thought of as a transient response, the randomdec signature can be used to provide the input needed by the moving-block method.

## Data System Implementation of

### Combined Randomdec/Moving-Block Method

The combined randomdec/moving-block method has been implemented on the automated data-acquisition system by using three FLUPS which are under the direct control of the test engineer. The three subroutines, designated BDRDIN, BDRD, and PKPLOT, are executed serially in the order listed and communicate with one another through labeled common blocks within the OMP program. BDRDIN is used to initialize some parameters; BDRD is used to calculate the randomdec signature; and PKPLOT is used to perform the moving-block analysis to determine the damping and frequency. The GDU is important in the execution of BDRD and PKPLOT. In the BDRD case, a plot of the signature is presented for the engineer's examination so that he can make a judgment as to whether or not the signature is satisfactory and warrants further processing by PKPLOT. In the execution of PKPLOT a large amount of interaction by the test engineer is required. This interaction includes not only the examination of data plots for their acceptability but also requires the engineer to input parameters to be used during the execution of PKPLOT. An intelligent choice of parameter values cannot be made without the engineer being able to examine plots of intermediate results on the GDU screen. The highlights of each of these three FLUPS are discussed in the following three sections.

FLUP BDRDIN. This subroutine essentially initializes a set of parameters to be used by the other two routines. The test engineer communicates with the FLUP through one of the keyboard input devices, usually the graphic display keyboard. Once the FLUP is initialized by typing in the FLUP name, the test engineer responds to queries displayed at the bottom of the GDU screen to input the required parameters. Initially, there are seven parameters required. These parameters are data channel to be analyzed, digital sampling rate, total number of data points to be sampled, number of data points in final randomdec signature, number of ensemble averages, and two parameters associated with converting the digital randomdec signature back to an analog signal for display on a recording oscillograph. The seven parameter values are stored in a labeled common block so that they can be accessed by the other two FLUPS. All seven of

these parameters are needed by BDRD, whereas PKPLOT only needs the sampling rate and the number of points in the signature. Once these seven parameters have been initialized, it is not necessary to execute BDRDIN again unless the engineer wishes to change a parameter, for example, increase the number of ensemble averages. Upon each call to BDRDIN, a table of parameter values selected is tabulated on the line printer to provide a permanent record.

FLUP BDRD. The primary purpose of this subroutine is to generate the randomdec signature. This FLUP must be executed by using the GDU keyboard. Initially, the subroutine calculates the mean and root-mean-square values of the output signal from a dynamic response sensor mounted on the model, usually a strain gage or an accelerometer. This information is used by the engineer to select a trigger level for the randomdec signature ( $Y_0$  value shown in fig. 8). Once the engineer has input the desired trigger level to the subroutine by using the GDU keyboard, the model response signal is digitized and passed to the computer for ensemble averaging. Typically, this process takes 1 to 2 minutes, depending on the number of data points in each time segment and the number of averages requested. Upon completion of the averaging process, the engineer is queried by a message on the GDU screen about the type of output he desires. He may select any combination of four options: plot signature on GDU screen, tabulate signature on line printer, plot signature using line printer, or convert digitized signature to analog signature and record on recording oscillograph. Normally, the engineer will select the GDU plot option so that he can examine the character of the randomdec signature. A randomdec signature plot is presented in figure 10. It should be noted that this figure and the subsequent figure that will be used in the PKPLOT discussion were prepared by using hard copy digital plots made by saving on digital magnetic tape the GDU plot vector file. (Such plotting is done after the wind-tunnel test run is completed by operating the computer in the batch mode.) Consequently, the GDU screen display was essentially the same as that shown in figure 10. Note the two computer output messages concerning the type of output desired and the engineer input message 0010 where he selected the GDU plot option in the GDU message area at the bottom of the figure. Since there is no absolute criteria as to what is or is not an acceptable signature, the engineer must be able to examine randomdec signature plots such as shown in figure 10 and decide, based on his judgment and experience, whether the signature is satisfactory. If the signature is not acceptable, the engineer will call BDRDIN and change some or all of the parameters, most commonly the number of ensemble averages, and then repeat BDRD. If the signature is acceptable, the engineer will call PKPLOT to determine the damping.

FLUP PKPLOT. The moving-block procedure is implemented in subroutine PKPLOT which is the most complex of the three subroutines and requires a considerable amount of interaction on the part of the test engineer. Initially, the subroutine accesses the randomdec signature that has been stored in labeled common by BDRD and plots the randomdec signature on the lower left-hand quadrant of the GDU screen as shown in figure 11(a). This plot is essentially a repeat of the plot that can be made by BDRD except in

BDRD the plot covers the entire GDU screen. Once the signature has been plotted, the subroutine uses a fast Fourier transform algorithm to determine the transform of the signature and displays the results in the upper left-hand quadrant of the GDU screen. The GDU screen now appears to the engineer as shown in figure 11(b). Note that the signature plot is retained in the lower left-hand quadrant. At this point the engineer can examine a portion of the Fourier transform in more detail if he desires by using the GDU keyboard to input a frequency range of interest to truncate and expand the frequency scale. The resulting truncated transform is displayed in the lower right-hand quadrant. The GDU screen now appears as shown in figure 11(c). For the example shown in the figure, a frequency range from zero to 50 Hz was selected. The GDU light gun now is enabled so that the engineer can select the frequency of one of the modes of the system, indicated by the peaks in the transform, so that the damping in that mode can be determined. The selected frequency is then optimized, and the moving-block method is applied to the randomdec signature by taking discrete Fourier transforms of the signature at this frequency. The natural logarithm of the transform amplitude is plotted against time in the upper right-hand quadrant on the GDU screen as shown in figure 11(d). It is from this variation with amplitude that the damping is obtained. Notice that there is a more or less straight-line portion near the beginning of the amplitude plot and then an oscillation in the amplitude appears. Such oscillations indicate that the mode being examined has damped out, and the calculations are being influenced by noise and other nearby modes. In some instances large oscillations occur throughout the entire amplitude plot. This is usually caused by too small a block size. A further clue to block size problems is indicated by the optimized frequency being considerably different from the selected frequency. If the amplitude plot does not appear to be satisfactory to the engineer, he can increase the block size by a keyboard entry and determine a new amplitude plot. The amplitude plot shown in figure 11(d) is a satisfactory one. The plot also indicates the need for engineering judgment in determining what portion of the curve should be used in calculating the damping. The engineer makes this judgment and inputs to the computer through the GDU keyboard the start and stop times (zero and 0.200 second were selected for the case shown) for a linear least-square fit to be applied to the amplitude data. This straight-line fit is displayed on the plot as shown in figure 11(d), and the engineer can accept or reject the fit. For the case shown in the figure, an acceptable fit was obtained. However, in this case the engineer would probably want to make some additional fits using different start and stop times since the logarithmic amplitude curve seems to indicate some nonlinear damping characteristics. One choice he would probably make is start and stop times of zero and 0.100 second, respectively. If the fit is rejected, he selects new start and stop times, and repeats the process until a satisfactory fit is determined. If the fit is acceptable, the damping is determined from the slope of the linear fit formula. The damping and frequency are displayed at the bottom of the GDU screen. At this stage in the process, the engineer may want to determine the damping in another mode. He can do this by selecting another input frequency and repeating the frequency optimization and discrete transform process to obtain a new amplitude plot from which the damping can be determined.



It should be pointed out that if the model is equipped with a means for producing a transient response as would be the case with an active control flutter model, an option is provided in PKPLOT for handling this case without first generating a randomdec signature. This amounts to using the moving-block technique alone. Although no applications of this type have been made to flutter tests, PKPLOT alone has been successfully used in some rotorcraft aeroelastic model studies.

#### Randomdec/Moving-Block Results and Discussion

The combined randomdec/moving-block method has been applied during several wind-tunnel flutter model tests in the transonic dynamics tunnel. One application was to the low-aspect-ratio arrow-wing configuration shown mounted in the wind tunnel in the photograph (fig. 12). These model investigations and companion analytical studies are part of a program to provide basic flutter data on candidate supersonic cruise aircraft configurations. Some subcritical damping results for the arrow-wing configuration are presented in figure 13 in the form of the variation of damping in the critical flutter mode with dynamic pressure and Mach number. It was necessary to plot the damping against both of these parameters since both were being varied as the flutter boundary was approached in a fashion similar to that shown in figure 7. Since this model was actually taken to the flutter condition, it is possible to get a direct comparison of the actual flutter condition and the condition that would have been predicted by extrapolating the subcritical damping measurements. The actual flutter condition is marked by the square symbols on the figure. Note that the extrapolated damping data predict a flutter condition very close to that actually found. The results shown here for the arrow-wing model are similar to those that have been obtained for several other configurations.

Although the combined randomdec/moving-block method is still in a developmental stage and has not yet completely replaced the traditional method of actually determining flutter points in defining the flutter boundary, it is beginning to prove useful in guiding model tests and preventing unexpected, sudden penetrations of the flutter boundary. Results obtained to date indicate that it is reasonable to expect that this method can be developed into a reliable subcritical damping method that will make it possible in many instances to define the flutter boundary without actually obtaining flutter. However, it has become obvious through the applications to date that a key ingredient to the successful application of the technique is the ability of the test engineer to exercise his judgment and experience by being able to interact in near real time by using the graphic display unit.

#### CONCLUDING REMARKS

The computer-controlled data-acquisition system recently installed for use with the Langley transonic dynamics tunnel has been described. This description has included a discussion of the hardware/software features of the system.

A subcritical response damping technique for use in real-time wind-tunnel model flutter testing that has been implemented on the data-acquisition system has been described in some detail. This technique is called the combined randomdec/moving-block method and its successful application requires that the user be able to interact with the computational process. The graphic display unit has proven to be a useful tool in providing plots of intermediate computational results so that the user can assure not only himself that the process is proceeding satisfactorily but also provides him with needed data so that he can intelligently select parameter values necessary to complete the damping calculations.

#### REFERENCES

1. Cole, Henry A., Jr.: On-Line Failure Detection and Damping Measurement of Aerospace Structures by Random Decrement Signatures. NASA CR-2205, 1973.
2. Hammond, Charles E., and Doggett, Robert V., Jr.: Determination of Subcritical Damping by Moving-Block/Randomdec Applications. Paper presented at Symposium on Flutter Testing Techniques, NASA Flight Research Center (Edwards, California), Oct. 9-10, 1975.
3. Johnston, J. F., and Conner, F.: The Reactionless Inplane Mode of Stiff-Inplane Hingeless Rotors. Rep. LR 26214 (Contract No. DAAJ01-73-O-0286), Lockheed-California Co., Dec. 1973.
4. Anderson, William D.: Investigation of Reactionless Mode Stability Characteristics of a Stiff Inplane Hingeless Rotor System. Preprint No. 734, American Helicopter Soc., May 1973.

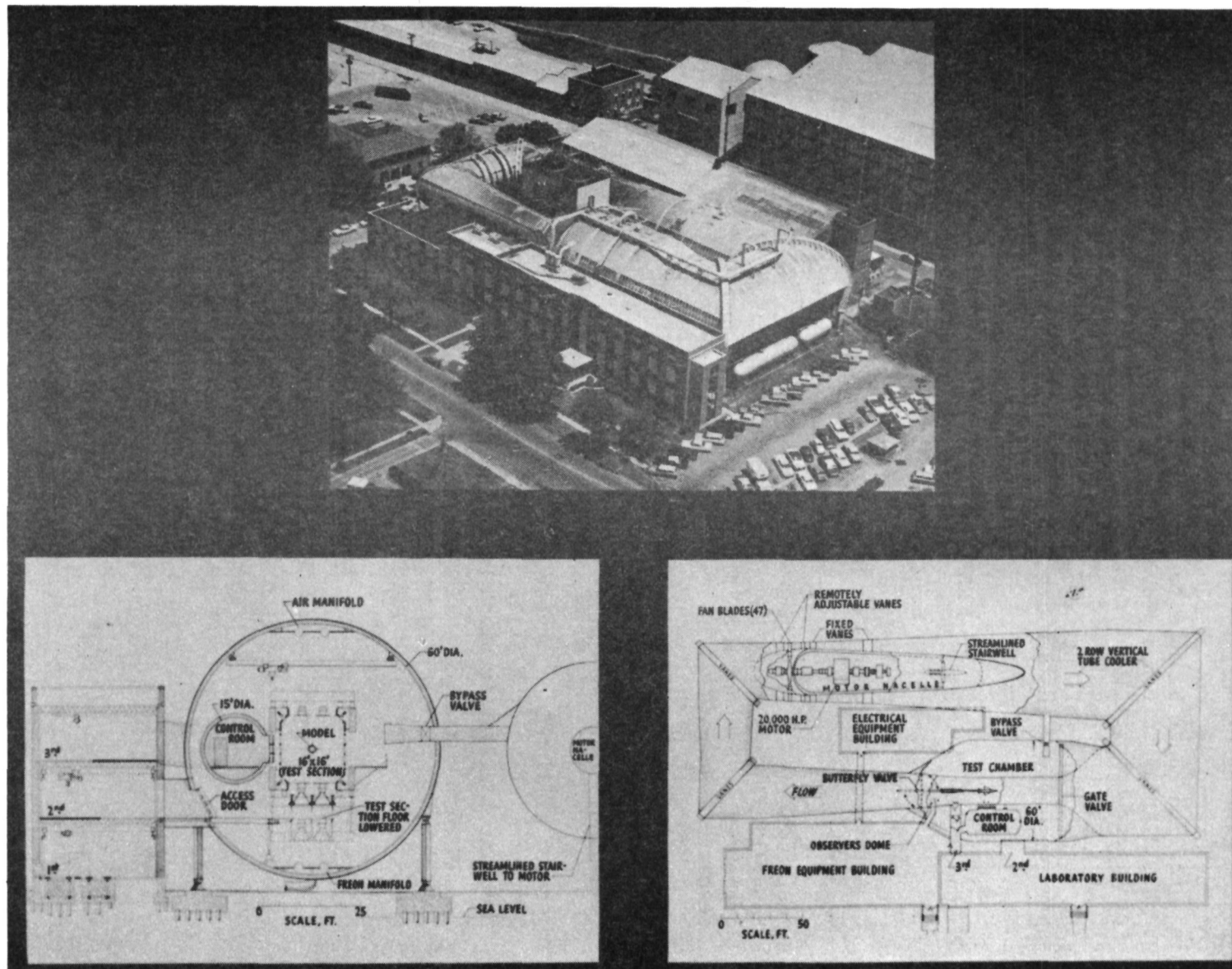


Figure 1.- Langley Research Center transonic dynamics tunnel.

TYPE OF TEST	MEASURAND	NUMBER OF CH	TRANSDUCER TYPE	FREQUENCY RANGE
BUFFET	ACCELERATION FLUCTUATING AERODYNAMIC PRESSURE	60	DYNAMIC PRESSURE CELLS PIEZOELECTRIC ACCELEROMETERS	5-20 KHZ
FLUTTER	STATIC FORCE ACCELERATION DYNAMIC FORCE MODEL PARAMETERS	50	STRAIN GAGE SERVO ACCELEROMETERS STRAIN GAGE ACCELEROMETERS	0-200 HZ
DECELERATOR	STATIC FORCE ACCELERATION DYNAMIC FORCE	13	STRAIN GAGES STRAIN GAGE ACCELEROMETERS	0-200 HZ
GUST LOADS	STRAIN ACCELERATION MODEL PARAMETERS	39	RESOLVERS STRAIN GAGES POTENTIOMETERS DIFFERENTIAL TRANSFORMERS PIEZOELECTRIC ACCELEROMETERS	0-50 HZ
GROUND-WIND LOADS	ACCELERATION MODEL PARAMETERS STATIC BENDING MOMENT DYNAMIC BENDING MOMENT	25	STRAIN GAGES POTENTIOMETERS SERVO ACCELEROMETERS STRAIN GAGE ACCELEROMETERS PIEZOELECTRIC ACCELEROMETERS	0-100 HZ
ROTORCRAFT	STATIC FORCE ACCELERATION DYNAMIC FORCE	32	STRAIN GAGES STRAIN GAGE ACCELEROMETERS	0-200 HZ
STATIC AERODYNAMICS	STATIC FORCE STATIC PRESSURE	48	STRAIN GAGES PRESSURE CELLS SCANNER VALVE CONTROLLER	0-5 HZ

Figure 2.- Types of wind-tunnel model tests and instrumentation requirements.

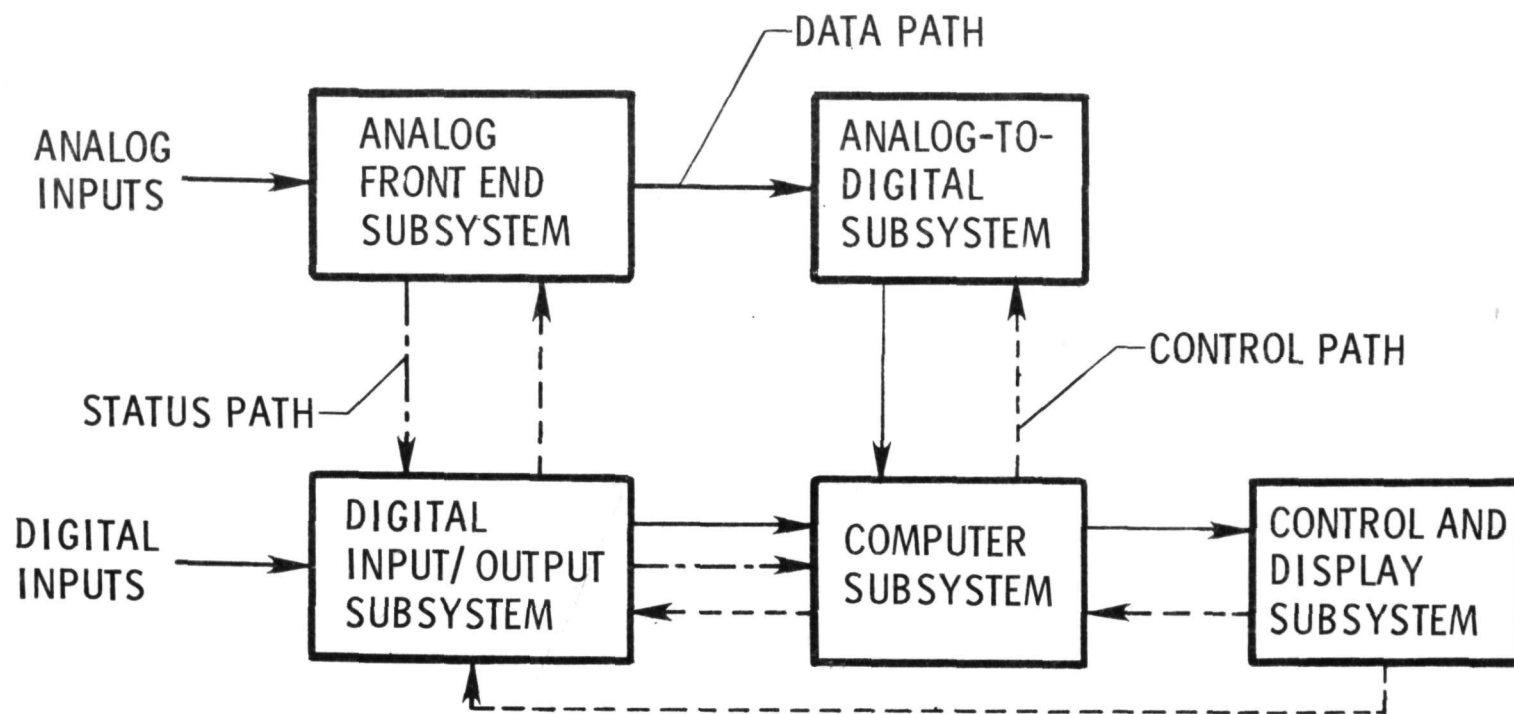


Figure 3.- Simplified block diagram of data-acquisition system.

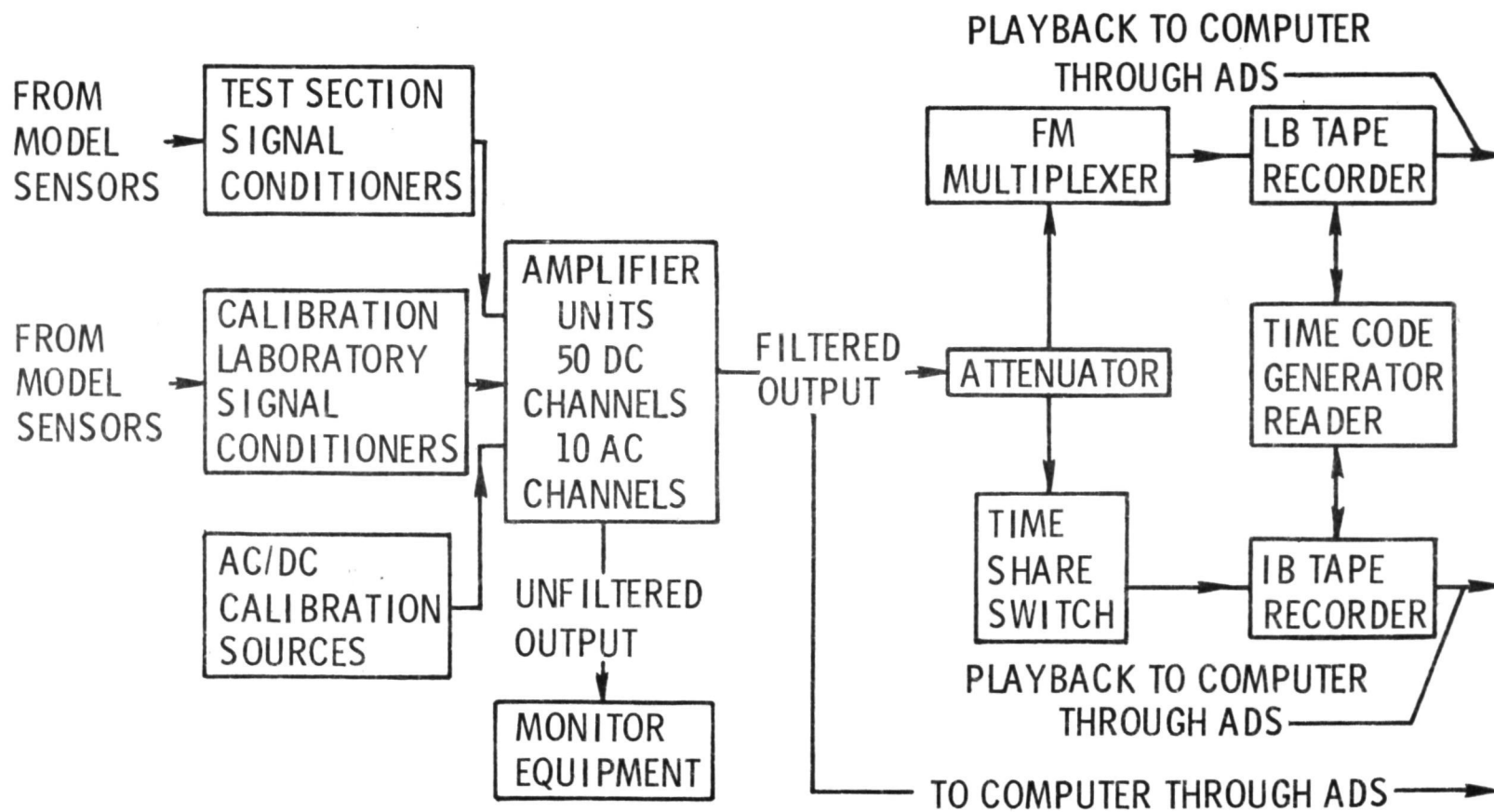


Figure 4.- Simplified block diagram of analog front end subsystem.

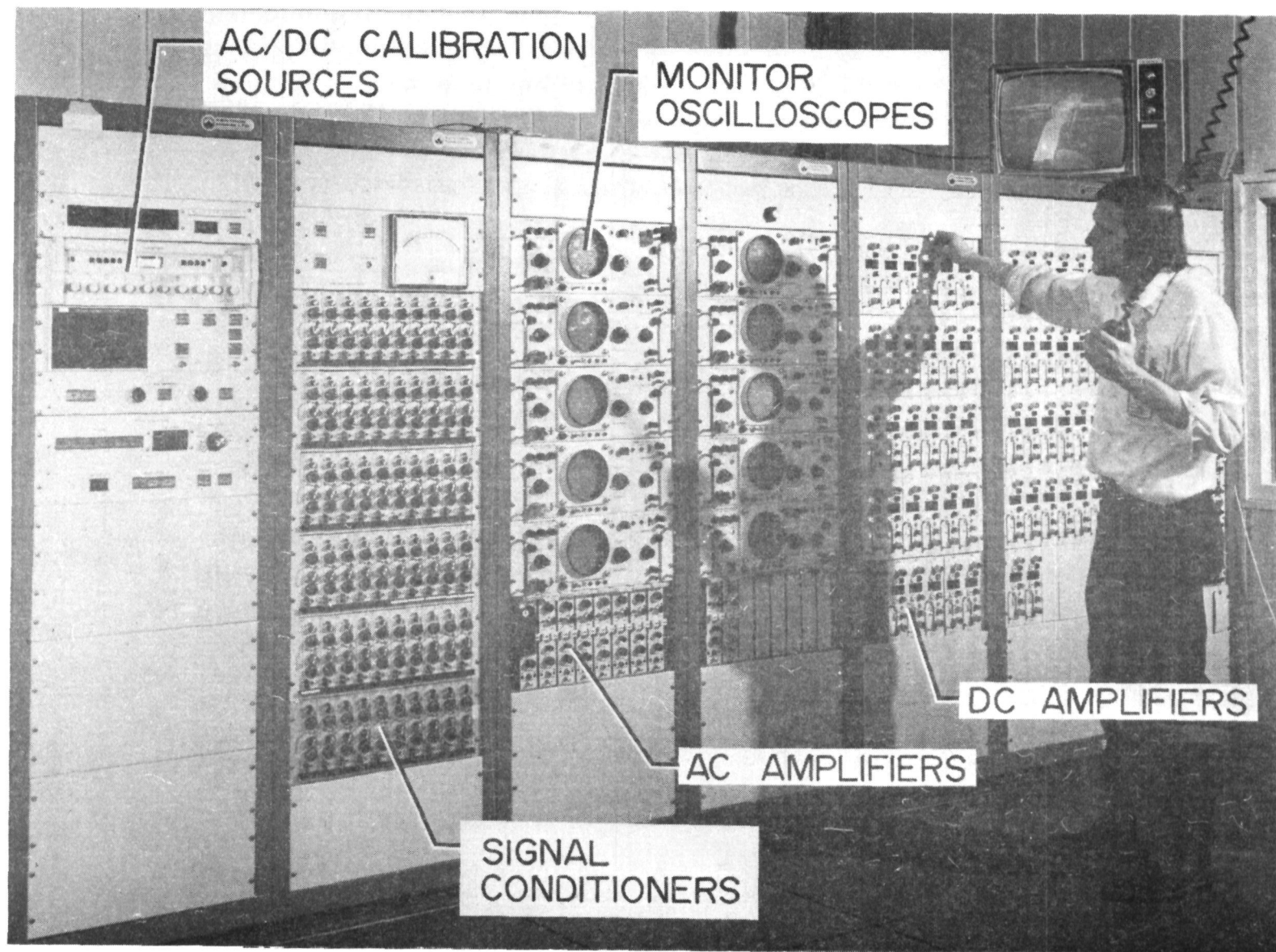


Figure 5.- Photograph of analog front end subsystem equipment.



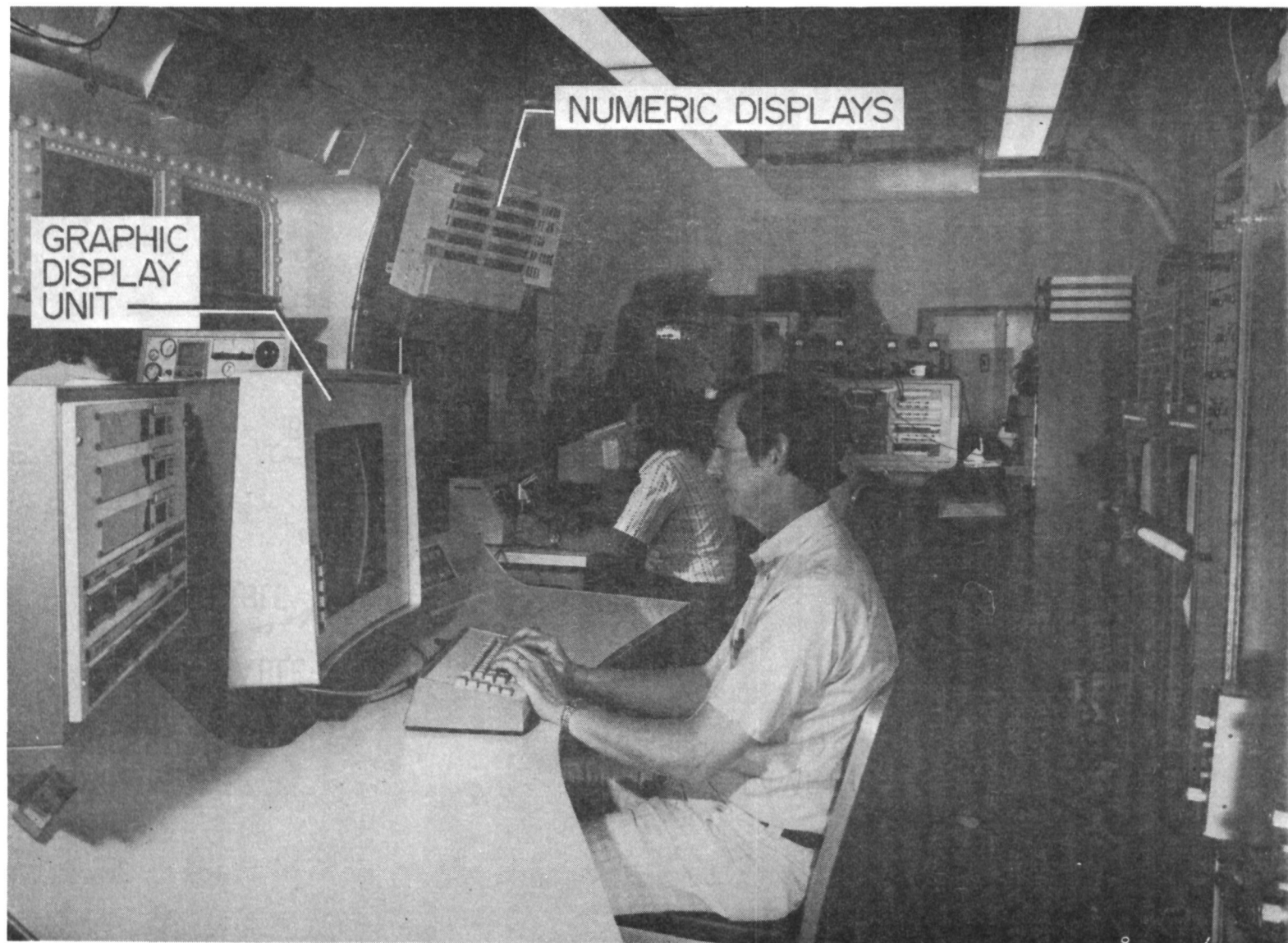


Figure 6.- Interior view of wind-tunnel control room.



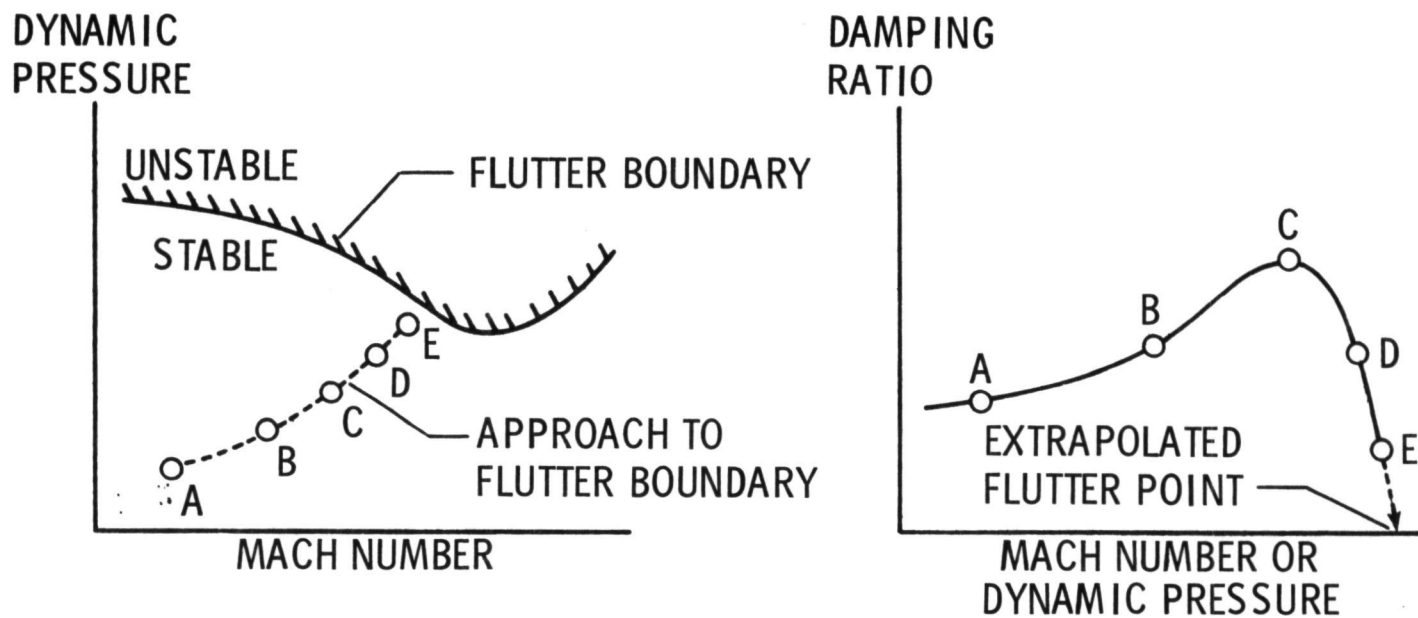


Figure 7.- Typical flutter boundary and subcritical damping variation.

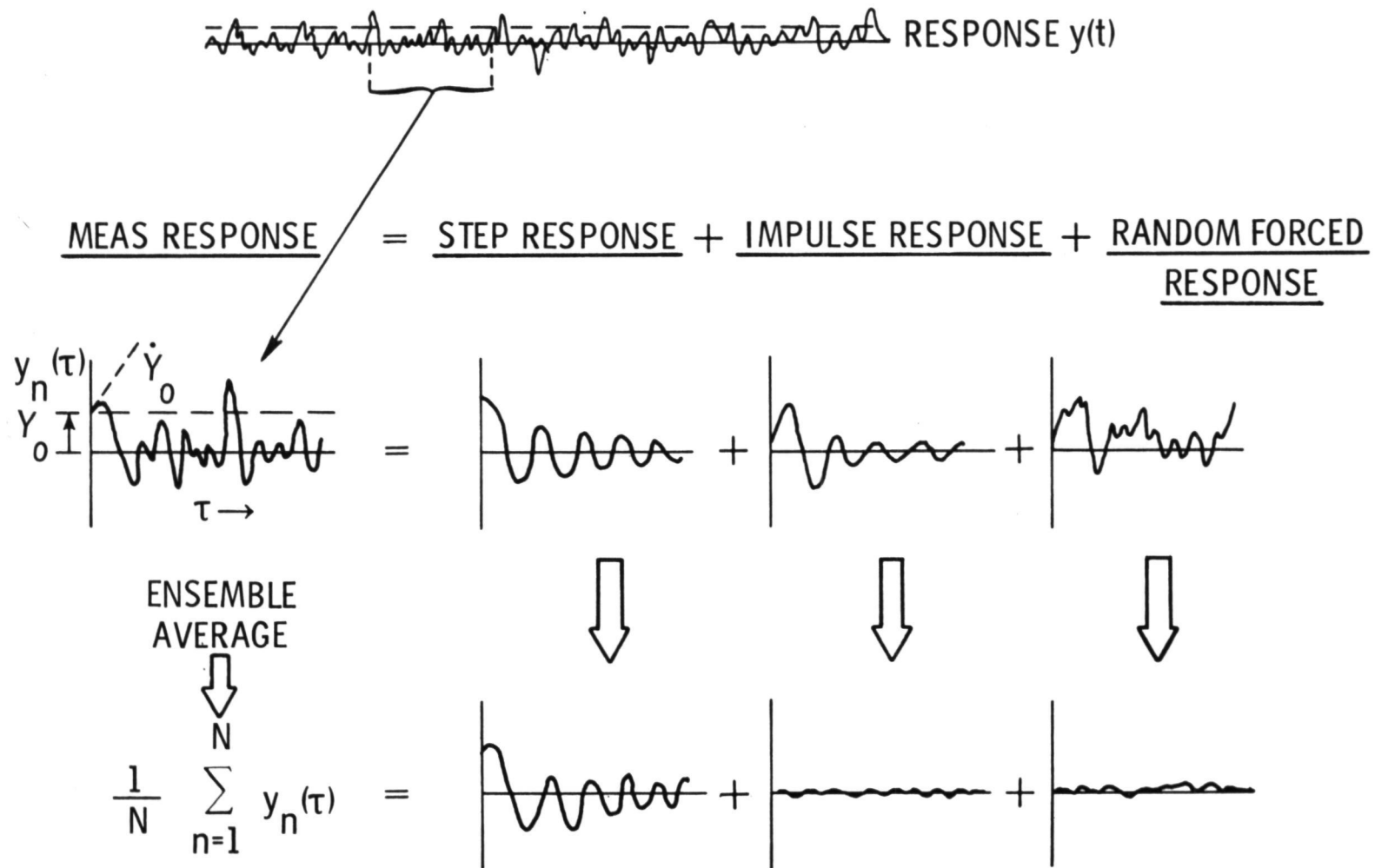


Figure 8.- Randomdec schematic illustration.

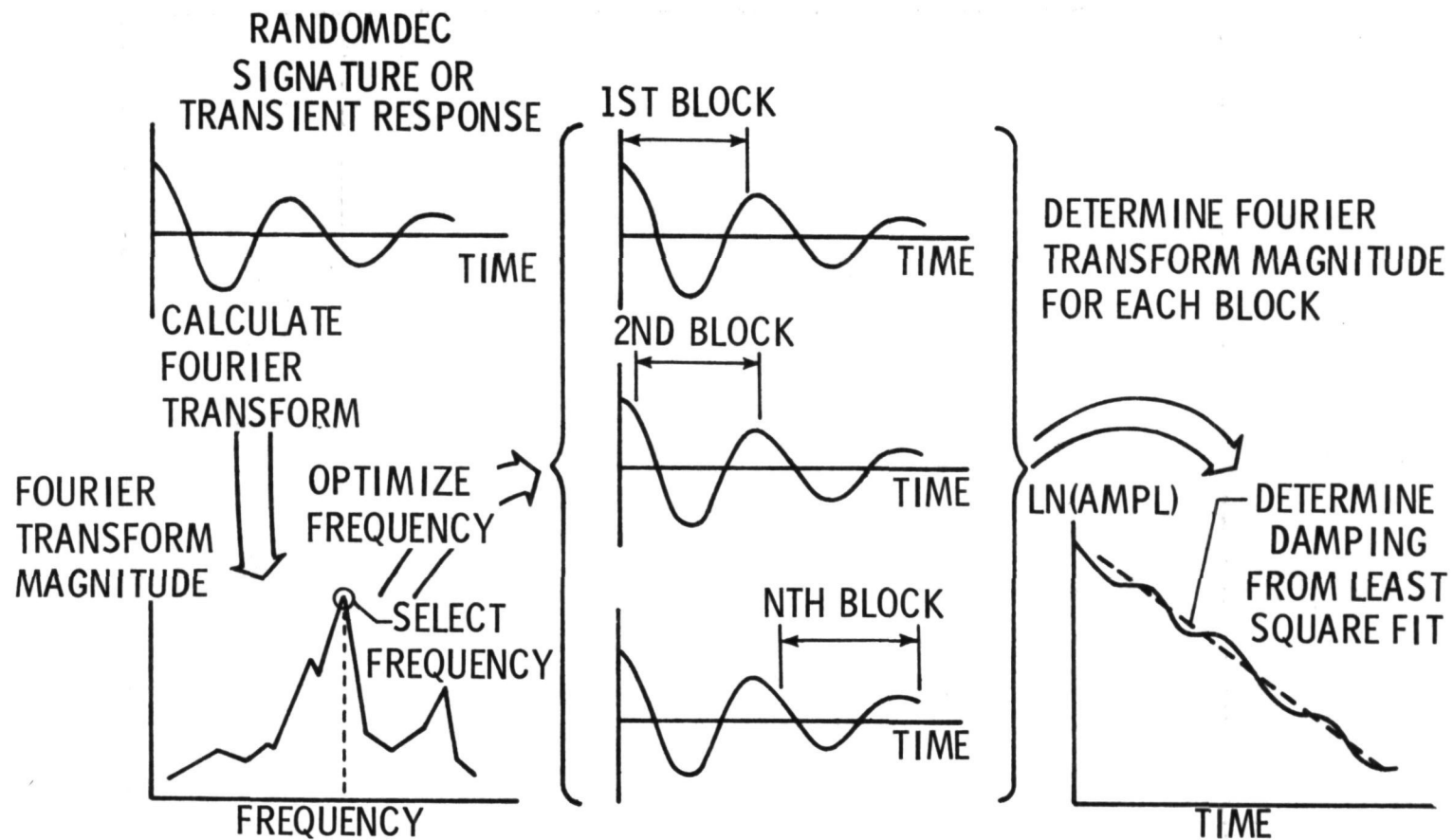


Figure 9.- Moving-block schematic illustration.

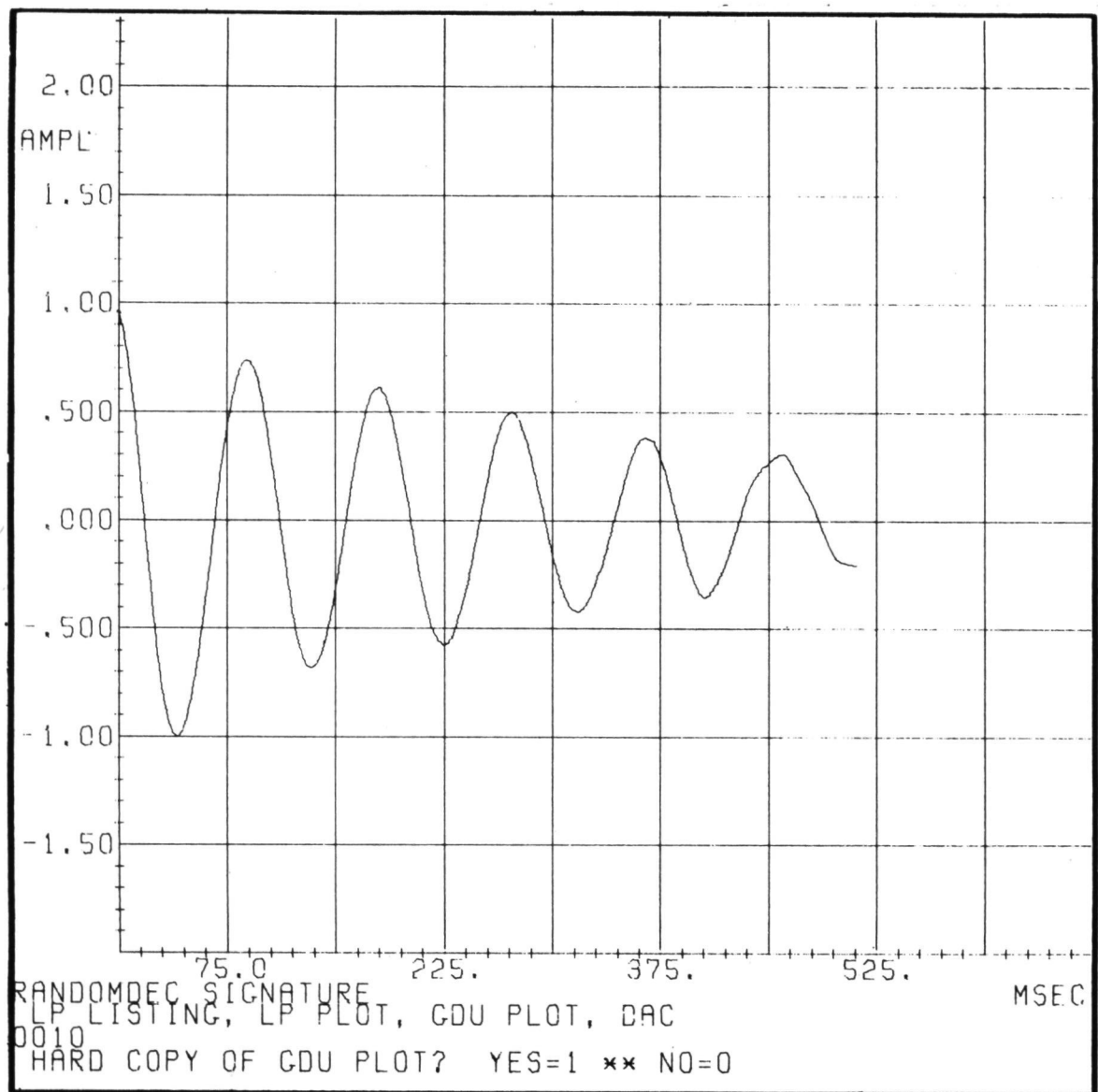
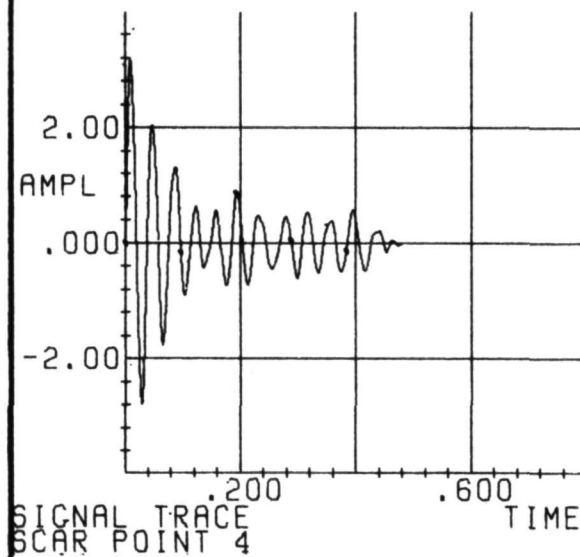
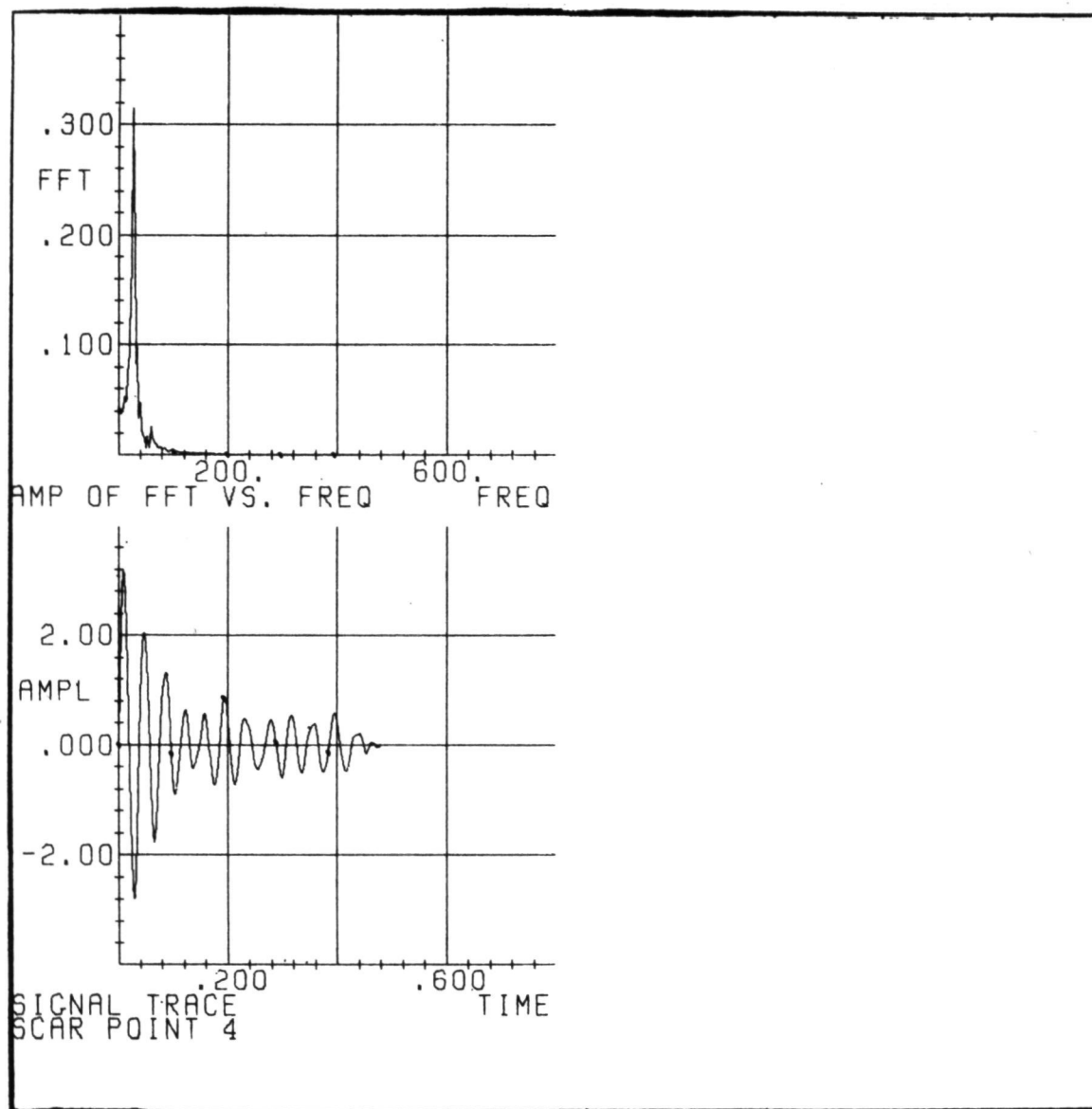


Figure 10.- Graphic display unit plot of randomdec signature.



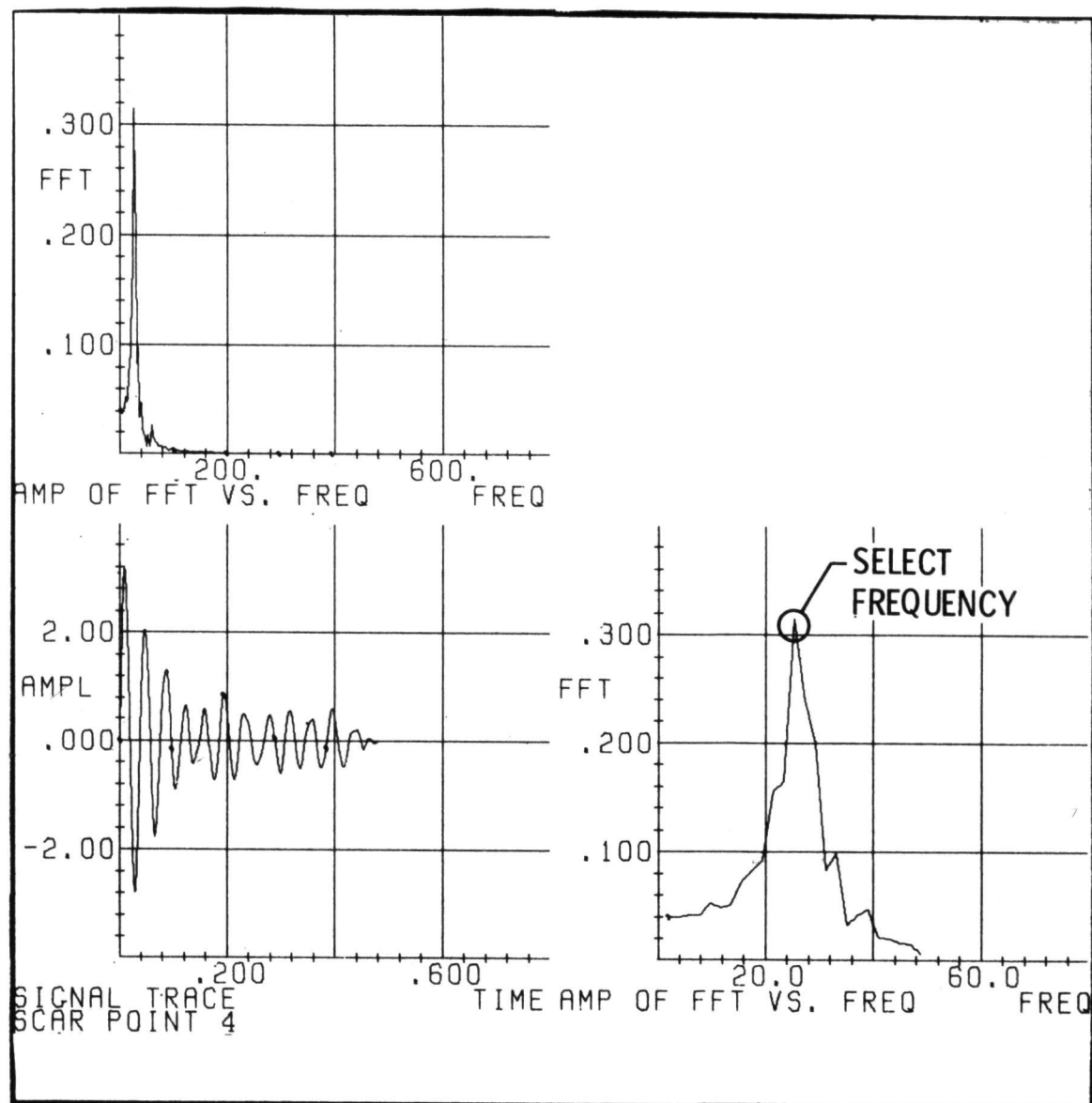
(a) Randomdec signature.

Figure 11.- Graphic display unit plot displays.



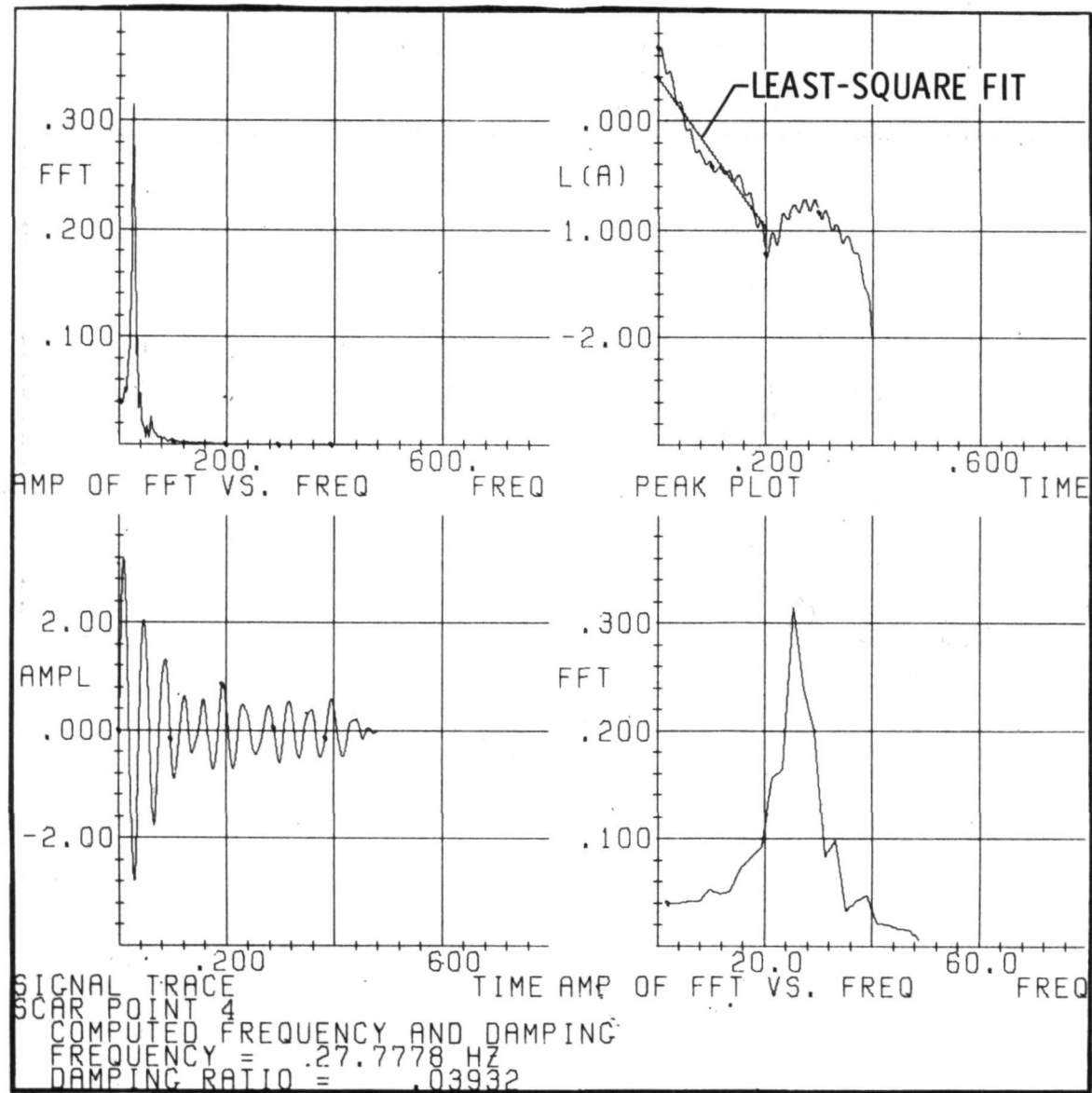
(b) Fourier transform amplitude.

Figure 11.- Continued.



(c) Expanded Fourier transform.

Figure 11.- Continued.



(d) Logarithmic amplitude.

Figure 11.- Concluded.





Figure 12.- Photograph of arrow-wing model mounted in wind tunnel.

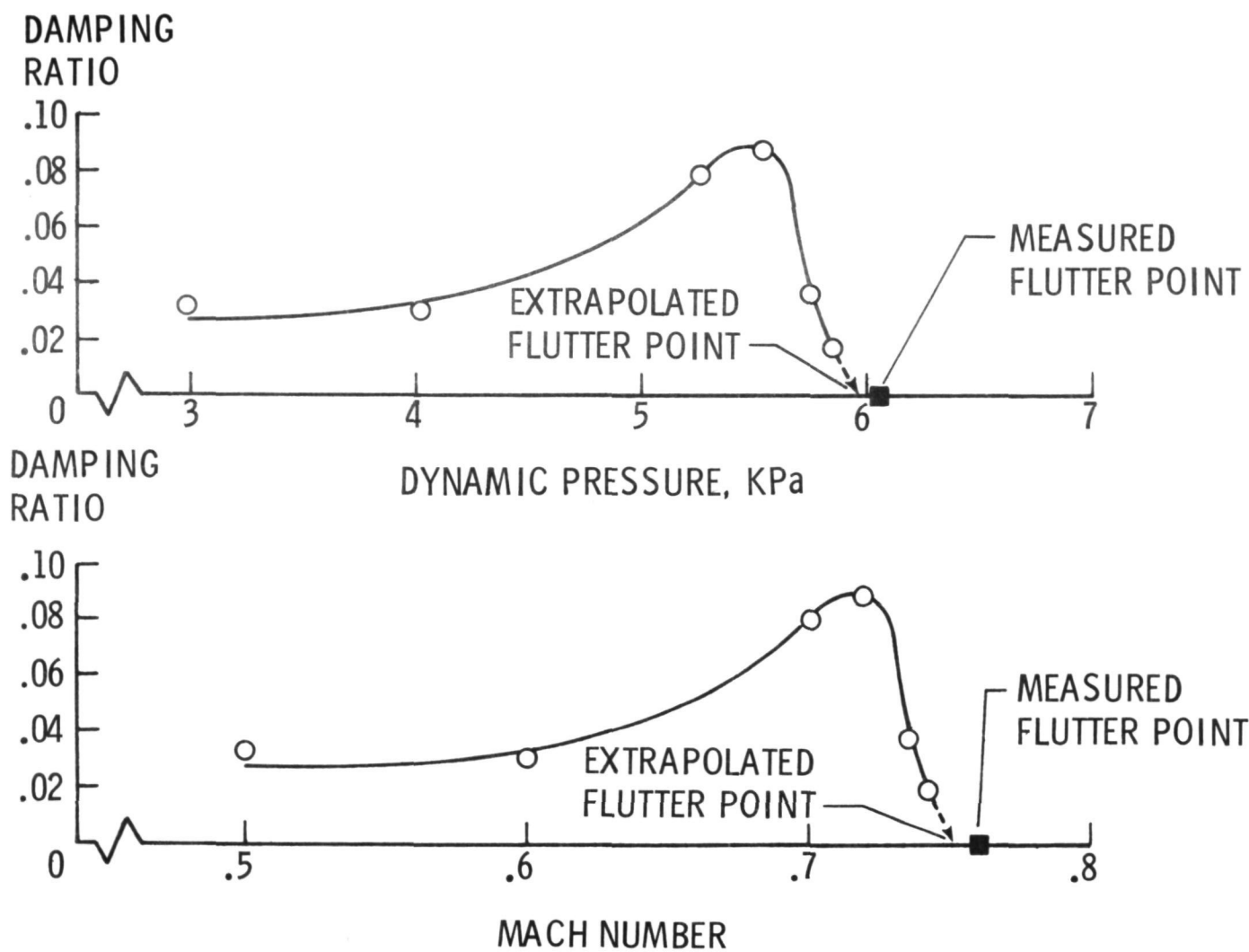


Figure 13.- Randomdec/moving-block subcritical damping results for arrow-wing configuration.

**Page Intentionally Left Blank**

## COMPUTER GRAPHICS IN ARCHITECTURE

### AND ENGINEERING

Donald P. Greenberg

Cornell University

### INTRODUCTION

Although the title which appears in the tentative program is "Computer Graphics in Architecture and Engineering", after hearing the first day presentations, it seems more appropriate to make some general comments. Thus, the content of this brief critique concerns three separate subject areas, related only by the fact that all are involved with interactive computer graphics.

### COMPUTER GRAPHICS IN THE ARCHITECTURE PROFESSION

Unlike the aerospace industry, the architecture or building profession is, unfortunately, completely fragmented. Structural, mechanical, and site engineers are usually separately contracted by the architect who is directly responsible to the owner-developer. Generally, the builder or general contractor is not actively involved in the design process until the successful acceptance of a bid. Although a few of these builders might perform their own "quantity take-offs", they usually subcontract large portions of the shop drawing preparation to their mechanical, electrical, or concrete subcontractors. These subcontractors, in turn, might even further subcontract so that the steel fabrication drawings or concrete formwork drawings are prepared separately.

This segmented building process as it exists today prevents the economic utilization of computer graphics. Without the ability to rely on a common data base, the true potential of interactive computer graphics cannot be realized. As a result, much of the publicized work has been primarily involved with the implementation of an efficient drafting system, but is not "computer-aided design" in the real sense.

In order for the potential for interactive computer graphics to be realized, it will be necessary to de-emphasize the re-creation of drawings, and to concentrate on the effective use of a three-dimensional data base.

Only then can the structural, cost-estimating and energy analyses be efficiently automated, and as a by-product, the working drawings also produced. However, the implementation of this logical process depends, not on any technological breakthrough in interactive computing, but primarily on the ability to vertically integrate the construction process.

#### CORNELL'S PROGRAM OF COMPUTER GRAPHICS

Cornell University has recognized the far-reaching importance of the emerging computer graphics technology in many aspects of scientific research and design. The University has established a Program of Computer Graphics and has received substantial support from the National Science Foundation for:

- (1) the development of computer graphics techniques,
- (2) the utilization of these techniques to solve specialized and varied research problems, and
- (3) the improvement of interactive design methodology.

A substantial community of faculty, staff, and researchers have been assembled to generate a broad, interdisciplinary research thrust in this direction. The cross-fertilization of specialists involved in many diverse problem areas provides a unique opportunity for demonstrating the unexploited potential of computer graphics. Some of the examples of research presently underway at Cornell which, in one way or another, maximize or require the use of computer graphics techniques follow:

- (1) Through data received from ground-based radar, computer graphics techniques will allow for a graphic depiction of the topography of Mars and Venus.
- (2) The pollution analysis of lakes can be greatly advanced by two- and three-dimensional pictorial real-time displays of the spread and effect of pollutants. It is possible to predict and graphically generate the lake circulation patterns using finite element techniques.
- (3) The use of interactive graphics capability in geological sciences is assisting in the determination of the fault mechanisms in the zone where two plates collide based upon the spatial distribution of deep earthquakes. The data being used is already available through research being conducted near the Tonga Islands by Cornell University.

- (4) Cornell scientists involved in the space exploration of the planet Jupiter wish to see the complex, time-dependent geometry of the plasma and magnetic fields surrounding that planet. The graphic visualization of these magnetic phenomena, which are extremely difficult to envision, could influence the selection of future flight paths.
- (5) The advantages of interactive computer graphics in structural engineering, particularly finite element analysis of complex geometric shapes, are obvious and have been adequately discussed at this conference. We are using these techniques for complete graphic input, including algorithms for mesh generation, optimum nodal numbering schemes, and displaying the results.
- (6) Architecture, since the entire discipline is founded on spatial representation, can benefit substantially from interactive computer graphics technology. At the present time, we have implemented a graphics and analytic capability to perform certain functions of automated structural design, cost-estimating, and energy analysis for the building sector.

These are only a few of the many existing research applications using interactive computer graphics at Cornell. There are many more areas, including water resources and flood control, protein synthesis, archeology, demography, and animation where efforts using computer graphics are currently being formulated, and where similar benefits would occur. Fortunately, since the Program is a "technique in search of a problem", I have had a unique opportunity to find out how truly diversified Cornell University really is.

Our present laboratory system includes a Digital Equipment Corporation PDP 11/50 with disk and tape units, an Evans and Sutherland Picture System, a Tektronix 4014 with hard copy, a Versatek printer/plotter, several digitizers, and a link to the University's IBM 370-168.

The laboratory presently has the capability of generating dynamic, black and white, wire-line drawings or perspective images of two- or three-dimensional objects. Emphasis is clearly placed on the utilization of both graphic input and output devices. It has the capability of interactive graphical input as well as hard-copy plotted output. Photographic equipment for both film-making and single-image documentation is included.

We are presently operating 22 hours/day, 7 days/week, and thus we are changing to a multi-user system. Since we have a need for four- and five-dimensional displays, we expect to have color display capability by the end of the academic year. A frame buffer will permit the assembly, using software, of the ordered information required to generate a static, continuous color image. A large MOS storage capacity combined with random access writing will

allow the researcher the opportunity to test many of the existing or new algorithms for half-tone picture generation.

#### CORNELL IN PERSPECTIVE: A COMPUTER SIMULATION

A fifteen-minute movie was made four years ago by twelve dedicated architecture students and me working at the General Electric Visual Simulation Laboratory in Syracuse. Since General Electric was on an eight-to-five shift, we worked from five-to-eight, three nights a week, for a semester. The movie was filmed from a standard television raster-display using a hidden-line algorithm requiring substantial preprocessing on object space. The maximum number of colors was limited to sixty-four appearing on any single image, and thus neither edge smoothing nor smooth shading algorithms are included. The story depicts the chronological development of Arts and Science quadrangle of Cornell from 1865 to 1975.

#### GENERAL COMMENTS

More seriously, after listening to the presentations of the previous session, I have the following observations, or questions, relating to the current state of interactive computer graphics as it exists in the United States today:

- (1) What means can be found to improve communications between people working in graphics? Much of the significant research which is presently being conducted at laboratories and universities throughout the country is "discipline-specific." Gathered at this conference is a substantial segment of the aerospace industry, all of whom have recognized the commonality of their problems with respect to computer graphics. But the technology required in other disciplines is also quite similar. Some very sophisticated computer graphics software capability has been developed in chemistry, biology, pollution studies, artificial intelligence, and animation. Yet the medium for information dispersal is poor and the information transfer is minimal. In fact, at this very moment, many computer graphic experts are attending a computer graphics animation conference in New York City.
- (2) Why have computer science departments, in general, not recognized or accepted computer graphics or computer-aided design courses as part of their curriculum?

- (3) How can we have an industry which collectively represents more than a billion dollar investment in computer graphics and computer-aided design, and simultaneously have only a small number of universities producing less than a dozen trained graduate students in this area per year?

If we are ever going to improve our productivity and progress in this field, then the above questions must be answered first. I personally believe that computer graphics and computer-aided design have not yet reached their potential. Furthermore, the demand for trained graduate students in this particular field is inevitably going to increase. Thus, I strongly urge your support, both moral and financial, to encourage universities throughout the country to educate students in this discipline.



**Page Intentionally Left Blank**

# COCKPIT DESIGN AND EVALUATION USING INTERACTIVE GRAPHICS

Susan M. Evans

University of Dayton Research Institute

## SUMMARY

This report presents a general overview of the characteristics of an interactive graphics system which has been developed to assist cockpit engineers design and evaluate work stations. The manikin used in this COMputerized Bio-mechanical MAN-model (COMBIMAN) is described, as are provisions for generating work stations and assessing interactions between man and environment. The applications of the present system are explained, and critiques of COMBIMAN are presented. The limitations of the existing programs and the requirements of the designers necessitate future revisions and additions to the biomechanical and ergonomic properties of COMBIMAN. Some of these enhancements are discussed.

## INTRODUCTION

During the design and analysis phases of work-station development, it is essential to assess the physical difficulties, inadequacy of conditions, or dangers of the work-station environment with respect to the human operator. The conventional method for accomplishing this has been to build mock-ups and then use an undetermined number of "representative" test pilots to evaluate the work environment and control placement. These mock-ups tend to be costly and time consuming to build, as well as inflexible during testing. The pilot sample size can range from one to 100, depending on pilot availability and the whims of the designers.

In an effort to assist in the design and analysis phases of work-station development, a COMputerized BIomechanical MAN-model (COMBIMAN) is being developed. It will serve as an interactive-graphics-assisted engineering tool to represent geometric, physical, and ergonomic properties of man at his work station. The tool will also aid in assessing interactions between man, equipment, and environment during task performance (reference 1). It will, therefore, eliminate the need for building mock-ups, as the designer can construct his work station in three dimensions on a Cathode Ray Tube (CRT) and can assess interactions by using man-models of various geometries.

The man-model used in COMBIMAN is based on a 30-link skeletal system, as shown in figure 1. The dimensions of the skeletal system can be altered by the user/designer. Since the link-lengths are generally internal and unmeasurable dimensions, link-lengths are based on 11 measurable anthropometric surface dimensions. The user can change the proportions of the model by specifying new values for any number of the surface dimensions. Using stored multiplication factors, the new internal link-lengths are then calculated. A similar relation-

ship has been developed between surface dimensions and link-widths and depths to assist in adding volume to the model.

The work stations designed and evaluated through COMBIMAN consist of three-to-six vertice panels, and control points located either on or off a defined panel. The more complicated work-station configurations developed to date consist of as many as 210 panels and well over 150 controls. All of the work stations which have been developed represent aircraft cockpit configurations, but it is possible to construct and display work stations of any type where operator interaction is essential. This would include automobile instrument panels, assembly line setups, and control panels for other types of military vehicles.

## METHODS

### Program Structure

The programs which comprise COMBIMAN have been written for use on an IBM computer. The machine which is presently used for the development of COMBIMAN is an IBM 370/155. All interaction with the IBM 2250 CRT is accomplished via IBM's FORTRAN-callable Graphic Subroutine Package (GSP). Interactive devices such as the 32 key Program Function Keyboard (PFK), the fiber-optic light pen, and the alphanumeric keyboard are also used. An overlaid structure of all the routines which make up the interactive package requires approximately 200 K bytes of memory.

The entire COMBIMAN system consists of five programs. Four of the programs are preliminary file creation/modification routines. The data generated by these preliminary programs are contained on an initialization data set, and anthropometric, task, and workspace data bases. The initialization data set contains constant link data used in assembling the man-model and the instructional messages displayed on the CRT during execution of program CBM04 (COMBIMAN, Version 4). The anthropometric data base consists of means, standard deviations and percentiles of surface dimensions from selected anthropometric surveys. The workspace data base and a task data base contain work-station configurations and task sequence information, respectively. The contents of these four files are accessed by CBM04, and their creation is essential prior to executing CBM04. A general data flow diagram of program CBM04 is shown in figure 2.

During the execution of the interactive graphics program CBM04, the user has a variety of options available to him through the use of the programmed function keyboard. The functions which have been implemented to date are shown in figure 3.

### Man-Model Generation

In order to display the man-model on the CRT, CBM04 uses information from both the initialization data set and the anthropometric data base, as well as user supplied data on a variable number of anthropometric surface dimensions

obtained at run time from the CRT. The ability to make use of user supplied dimension data permits the construction of man-models of variable proportions thus generating a man-model suited to the particular needs of the user. While seated at the CRT, the user can select a survey from the anthropometric data base, and then may choose to input all 11 anthropometric surface dimensions or two key dimensions related to height and weight. These dimensions can be supplied as percentiles, that is 5th, 95th, 50th, etc., or as absolute values. If only two key values were supplied, the balance of the dimensions are calculated based on stored regression equations. To obtain the needed link-lengths, the appropriate anthropometric dimensions are multiplied by predetermined factors stored in the initialization data set. These link-lengths, in conjunction with available link hierarchies and the angular relationship between connecting links, are used to generate the skeletal system of the man-model.

A man-model consisting of solely a link system would provide the user with insufficient information on necessary cockpit dimensions. Volume about the link system is necessary to give the user body support and control placement data. The skeletal system is enfleshed, or supplied with volume, by placing elliptical cylinders about crucial links. The dimensions of the major and minor axes of the proximal and distal ellipses of each link are derived from the multiplication factors stored on the initialization data set and from the relevant anthropometric dimensions used to generate the link lengths. A few special links are enfleshed by ellipsoids, while others, such as those connecting SRP and MID HIP, MID HIP and RIGHT HIP, and MID HIP and LEFT HIP are not enfleshed at all. Third degree polynomial equations have been developed to curve over joint centers, where necessary, to eliminate gaps (reference 2). A view of the enfleshed man-model can be seen in figure 4.

The procedure for generating the enfleshed man-model configuration is to fix to each link  $L_{I_i}$ , a local coordinate system  $C_{I_i}$ , at the distal joint of the link  $P_{I_i}$ , with  $Z_{I_i}$  - axis directed along the link in the distal direction. Figure 5 illustrates the local coordinate systems of two links,  $L_{I_i}$  and  $L_{I_{i-1}}$ .

The angular relationship between links can be expressed in terms of the transformation between the local coordinate systems. For example, in figure 5, local coordinate system  $C_{I_i}$  is related to local coordinate system  $C_{I_{i-1}}$  by a constant translation of  $L_{I_i}$  and a three-dimensional rotation. Thus, a positional vector  $\{R_{I_i}\}$  in  $C_{I_i}$  is transformed to a positional vector  $\{R_{I_{i-1}}\}$  in  $C_{I_{i-1}}$  by the matrix equation

$$\{R_{I_{i-1}}\} = T_{I_i} \left[ \{R_{I_i}\} + \{t_{I_i}\} \right] \quad (1)$$

Where  $\{t_{I_i}\}$  is constant vector in  $C_{I_i}$  with components  $[0, 0, L_{I_i}]$  which causes

the translation, and  $T_{I_i}$  is a rotation transformation matrix which represents the three-dimensional rotation of a local coordinate system. The best means of expressing this transformation is by the Euler angles phi ( $\phi$ ), theta ( $\theta$ ), and psi ( $\psi$ ), which are used commonly in rigid body dynamics. The three Euler angles used correspond to the three rotations of the coordinate axis of the  $C_{I_{i-1}}$  system to get to the position of the coordinate axis of the  $C_{I_i}$  system. The transformation matrix  $T_{I_i}$  for the coordinate system  $C_{I_i}$  is shown in figure 6. For the location of the joints, we chose as the reference frame a cartesian coordinate system with its origin at the base point  $P_0$ , z-axis directed upward, x-axis directed to the front of the man-model, and y-axis directed to the left side of the man-model. In order to obtain the joint locations, it is necessary to trace the angular relationship from the base point to the joint in question. This may be accomplished conveniently by the transformation matrix  $T_{I_i}$ . Instead of considering only the joint locations, we shall consider a more general case of how a position vector  $\{R_{I_i}\}$  in  $C_{I_i}$  transforms into a position vector  $\{R_{I_j}\}$  in  $C_{I_j}$ , where  $j < i$ . This is essential when trying to en flesh the link system. The transformation of  $\{R_{I_i}\}$  may be accomplished by the recursive application of equation 1. The general case can be deduced as

$$\{R_{I_j}\} = \left( \prod_{k=j+1}^{k=i} \right) \{R_{I_i}\} + \sum_{m=j+1}^{m=i} \left( \prod_{k=j+1}^{k=m} T_{I_k} \right) \{t_{I_m}\} \quad (3)$$

where  $j < i$ . Figure 7 shows the preferred transformation angle values for phi, theta and psi, and the link lengths used to position a 50th percentile man-model in seated erect position. The above equations represent the core of the COMBI-MAN program designated as CBM04. They are used to assemble the link system of the man-model, to position the enfleshment around it, and to position the enfleshed man-model within a work station. For a more detailed description of the geometry of the model see reference 3.

#### Work-Station Generation

Two methods are used to generate and display work stations, depending on whether the designer chooses to use an existing configuration, or decides to construct a new one on the CRT using the light pen. Panels and controls for existing configurations are stored on the workspace data base. Prior to running CBM04, the workspace data base maintenance program read in the coordinates of the vertices of each panel as well as the coordinates of each control and converted the points to the right handed coordinate system used throughout the

COMBIMAN system, with the origin located at seat reference point (SRP). Once an existing work station has been retrieved, it and the man-model are rescaled and displayed as two orthogonal views on the CRT.

The second method of generating work stations has the user designing a work station on the CRT with the use of the light pen, alphanumeric keyboard and PFK, following the basic series of steps similar to those used on a drawing board. These steps include locating SRP, specifying back rest and seat pan angles as well as dimensions, and determining line-of-sight and heel rest line of the seated operator. The program CBM04 has been designed to request operator information in a predetermined sequence. As shown in figure 8, three dimensions are projected onto the display area of the screen by using two, two-dimensional views (two orthogonal views) of the man-model and/or work station.

The designer can light pen points on the screen in three dimensions by following the following steps: 1) Position the tracking symbol with the light pen at a point in the left view (normally the X-Z plane of the image) and depress the appropriate PFK; this will signal the program to read the screen coordinates of the light pen. 2) As soon as a horizontal line is displayed on the right side (normally the Y-Z plane) at the Z-level established in the left side, position the tracking symbol with the light pen and again depress the appropriate PFK. This will cause the screen coordinates to be read and will supply the program with a third coordinate for the XYZ triple. Because the screen coordinates are scaled values, the values are converted back to real world units for the benefit of the user and for use in future calculations by the program. This basic sequence of steps is used repeatedly by the designer to construct panels, define controls, and to determine the location of points within the work-space.

### Reach Analysis

The key method of assessing interaction between the man-model and work station involves a reach analysis routine which is part of the CBM04 program. The purpose of the reach analysis routine is to determine whether a point in space can be reached by human operators of various anthropometric dimensions. The user light pens the point to be reached, and specifies the link on the man-model where motion is to start. Specifying the latter item allows the designer to restrict motion of certain body segments, as in simulating a restraining cockpit environment where the seated operator is unable to move his back. The routine places the most proximal link of the chain or series of links in a position that brings its distal end closest to the test point. A chain is defined as a series of connected links, such as those links used in the torso-head or right arm systems. This procedure is repeated for the remaining body links of the chain. The specified point is considered to be within reach if the distal point of the most distal link can be placed at that point. The positioning of an individual body link closest to the point to be reached is accomplished by using the IBM-supplied minimization subroutine DFMFP, which is based on the method developed by R. Fletcher and M.J.D. Powell (references 4 and 5). The objective function and gradient vector which are required by the DFMFP routine have been developed by University of Dayton Research Institute (UDRI) personnel (references 2 and 3).



The angular limits of mobility, the minimum and maximum values for each of the transformation angles, are obtained from the initialization data set and are used in calculating the objective function and gradient vector to avoid unrealistic positioning of the man-model.

## RESULTS

By using the man-model, work station and reach analysis routines mentioned above, the work-station engineer can gain additional information about the interactions of human operators and work-station environments.

The reach analysis routine explained above, in conjunction with the other elements of COMBIMAN, can assist the designer in determining static body positions of the model. A key element in this procedure is the joint mobility constraints. These angles, when used by the reach routine, prevent the model from assuming a humanly impossible position. Until the model includes data on link weight factors, the resulting static position may not be the position most likely assumed, but it will nonetheless be possible to assume it.

The reach analysis routine can also be used to establish reach envelopes. For this purpose, a point in space is light penned and the reach analysis is initiated. After the first attempt the point can be light penned again, this time at a location just beyond the reach capabilities of the man-model. The most distal point of the displayed man-model will then mark the point of maximum reach in that direction. The point to be reached can then be moved in increments in any desired direction. Each one of these moves results in the determination of another point of maximum reach. Thus reach envelopes can be point wise established on the CRT. By adding the user-initiated option of storing these points, they would be available at the time the user requested hard copy plots of the envelopes. Since changes in body geometries tend to alter the definition of the reach envelopes, the user could obtain envelopes for an infinite number of combinations of body geometries by altering the anthropometric dimensions used in generating the man-model.

By combining the above mentioned applications, the designer can analyze the present placement of major controls within the displayed work station with respect to the reach envelope of the seated operator, and he can determine their optimum location. He can also analyze the dimensions of the work station with respect to the body geometry of the operator. Controls and panels of predefined, as well as newly developed, work stations can be deleted and then redefined by following the same sequence of steps using PFK and light pen as used to design a new work station. The geometry of the man-model can be varied to fully evaluate the particular area being tested.

In assessing control placement, for example, man-models of the 5th, 50th and 95th percentiles of a particular survey may be displayed within a work station, one after another, and instructed to reach to a particular control. A printed message on the CRT, as well as the repositioned model, will assist the user in determining the correct control placement for this range of the population. In another example, when evaluating ejection seat clearance, a key anthro-

pometric dimension involved is Butt-Knee Length. The designer may change this dimension alone, or this along with Shoulder Breadth or any number of others, generating as many combinations of dimensions as he needs to fully analyze the clearance dimension requirements.

The standard COMBIMAN CRT display area consists of a prompting area, an informational area, and a display area, similar to that shown in figure 9. At any time during the design and analysis phases of COMBIMAN, the contents of the display area can be rotated and/or magnified to gain a more detailed and accurate view of the man-work-station combination. Both views may be rotated, or just one of the two can be enlarged to occupy the entire 12" screen, rather than the normal 6" square. Rotation of the display to any plane will provide the user with a complete three-dimensional composite of the configuration. To further assist the user, hard copy plots of the present configuration can be requested, as well as a printed copy of man-model and work-station coordinates. At termination of the program CBMO4, a detailed activity log is printed.

#### FUTURE RESEARCH

Additional research and redesign efforts are underway to eliminate the limitations of the present system. One of such limitations is the lack of interference handling. The mobility constraints of the man-model prohibit it from reaching through itself but there is no provision for avoiding interference of the work station itself. This includes control sticks, instrument panels or back rests. Another limitation was mentioned before. The static body positions as derived and displayed are humanly possible to assume, but in some cases, they would not be the position the majority of human operators would assume. Within the next few months, the body positioning of COMBIMAN will be improved to achieve a more realistic simulation of static body positions and to simulate dynamic body motions on the CRT. This incorporation of kinematic reach with typical movement patterns will be based on pertinent available experimental results, as well as segment mass distribution and moment of inertia data.

Enhancement of the ergonomic properties of COMBIMAN will enable the user to better assess man, equipment and environment interactions during actual task sequences. In addition to the incorporation of kinematic reach, scheduled additions include incorporation of ground visibility plots and the effects of personal-protective equipment and acceleration fields. These additions, when used in conjunction with the COMBIMAN model, will allow the user to specify a task sequence contained on the task data base, a specific G-force, encumbering equipment, and body supports, and then allow the user to watch as the model performs within the specified environmental conditions. This capability will provide the designer with information on safety, comfort and performance not easily measurable in real life.

#### CONCLUDING REMARKS

With the COMBIMAN system in its present state of development, the designer can display an existing work station from the workspace data base, or can generate one from scratch, or even combine parts of existing work stations with newly



designed ones. He can add a man-model with dimensions to suit his needs. He can then vary the position of the model and the configuration of the work station as many times as necessary to obtain the best possible work-station environment. To attempt to obtain this variability in man and work station with any other method than a computer drawing board would be impossible.

As the anthropometric, biomechanical and ergonomic analogs of COMBIMAN advance in development, COMBIMAN, as an engineering tool will increase in power. COMBIMAN, as an anthropomorphic dynamic analog of the man-cockpit interface, will be a powerful engineering tool in the evaluation of existing, or the planning and design of new manned systems. Its flexibility and power will serve the engineers' needs directly and will accelerate or eliminate the manual phases of design work.

#### REFERENCES

1. Kroemer, K.H.E.: COMBIMAN-Computerized Biomechanical Man-Model. AMRL-TR-72-16, 1973.
2. Dillhoff, K.J., Evans, S.M., and Krause, H.E.: Incorporation of Reach, Dynamic and Operational Capabilities in an Improved COMBIMAN Man-Model. UDRI-TR-74-50, 1974.
3. Bates, F.J., Evans, S.M., Krause, H.E., and Luming, H.: Three Dimensional Display of the COMBIMAN Man-Model and Workspace. UDRI-TR-73-47, 1973.
4. Fletcher, R. and Powell, M.J.D.: A Rapidly Convergent Descent Method for Minimization. Computer Journal, June 1963, pp. 163-168.
5. Fletcher, R. and Powell, M.J.D.: Function Minimization by Conjugate Gradients. Computer Journal, July 1974, pp. 149-154.

#### ACKNOWLEDGMENTS

Research for COMBIMAN by the University of Dayton Research Institute is sponsored by the Crew Station Integration Branch, Human Engineering Division, 6570th Aerospace Medical Research Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. The current effort is funded by Contract No. F33615-75-C-5092.

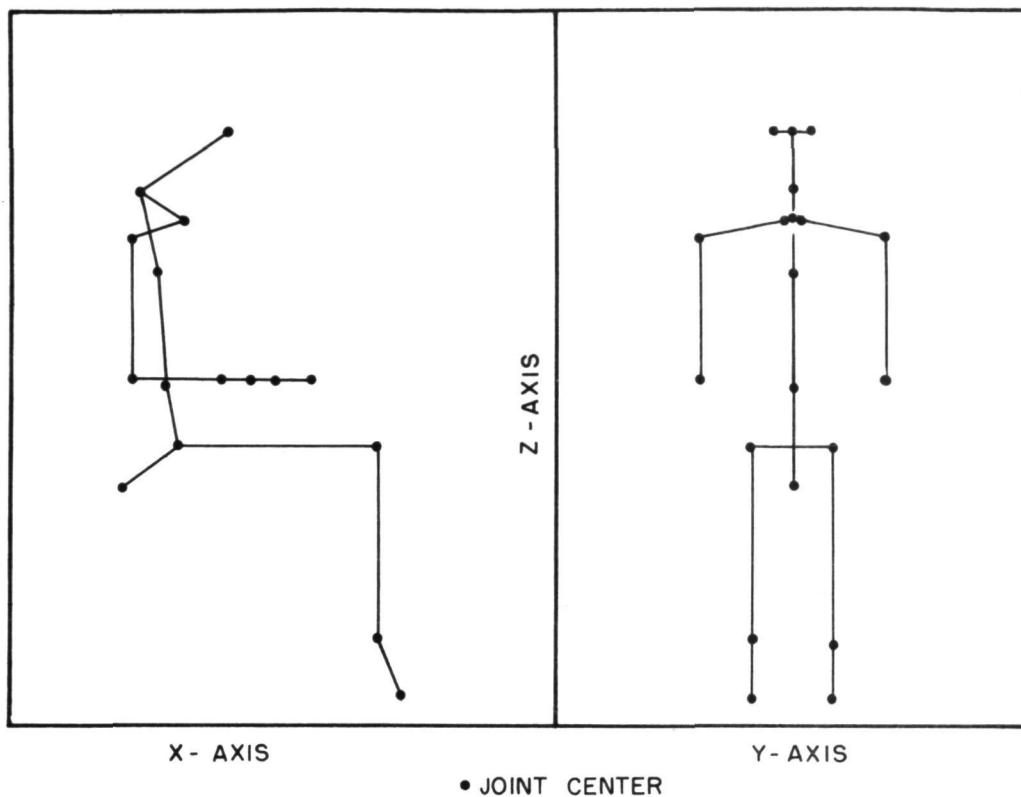


Figure 1.- Link system of present COMBIMAN man-model.

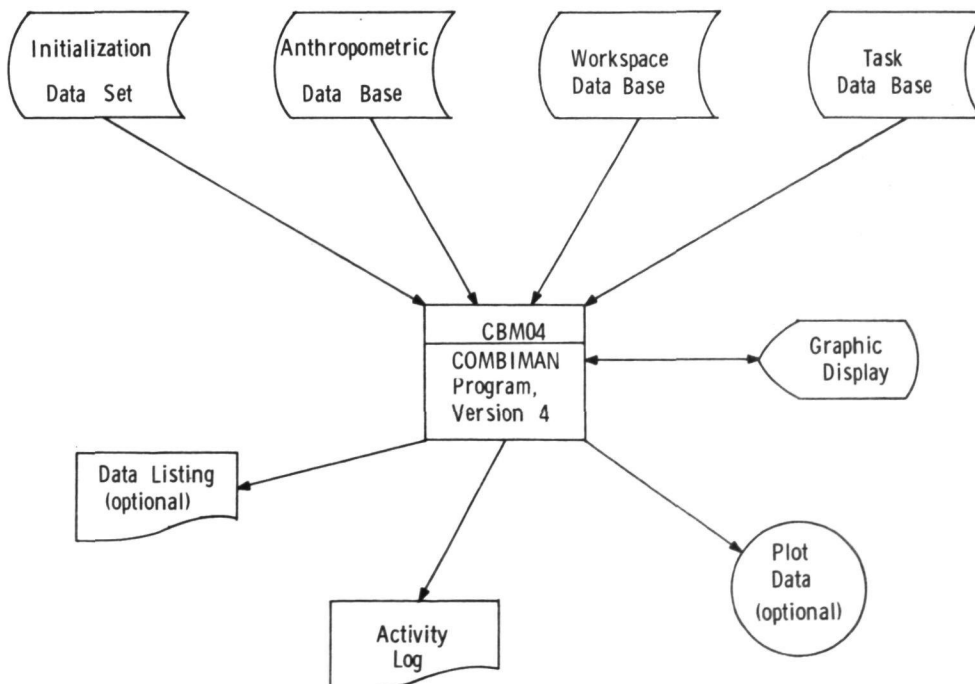


Figure 2.- Data flow in the COMBIMAN interactive graphics program CBM04.

ANTHROPOMETRY-RELATED

Retrieve Anthropometry  
Enter Link Dimensions  
Enter Two Key Dimensions  
Display Link Table

WORKSPACE-RELATED

Retrieve Workspace  
Design Panel  
Define Control  
Delete Panel  
Delete Control  
Change Panel  
Change Control

DISPLAY-RELATED

Change View  
Identify Object  
Omit Object  
Include Object  
Note Light-Pen Location

MAN-MACHINE-INTERACTION-RELATED

Perform Task  
Perform Reach

PRINTER/PLOTTER-RELATED

Print Data  
Plot COMBIMAN

PROGRAM-EXECUTION-RELATED

Set Sense Switch  
Restart Program  
End Program

Figure 3.- Program function keys implemented to date.

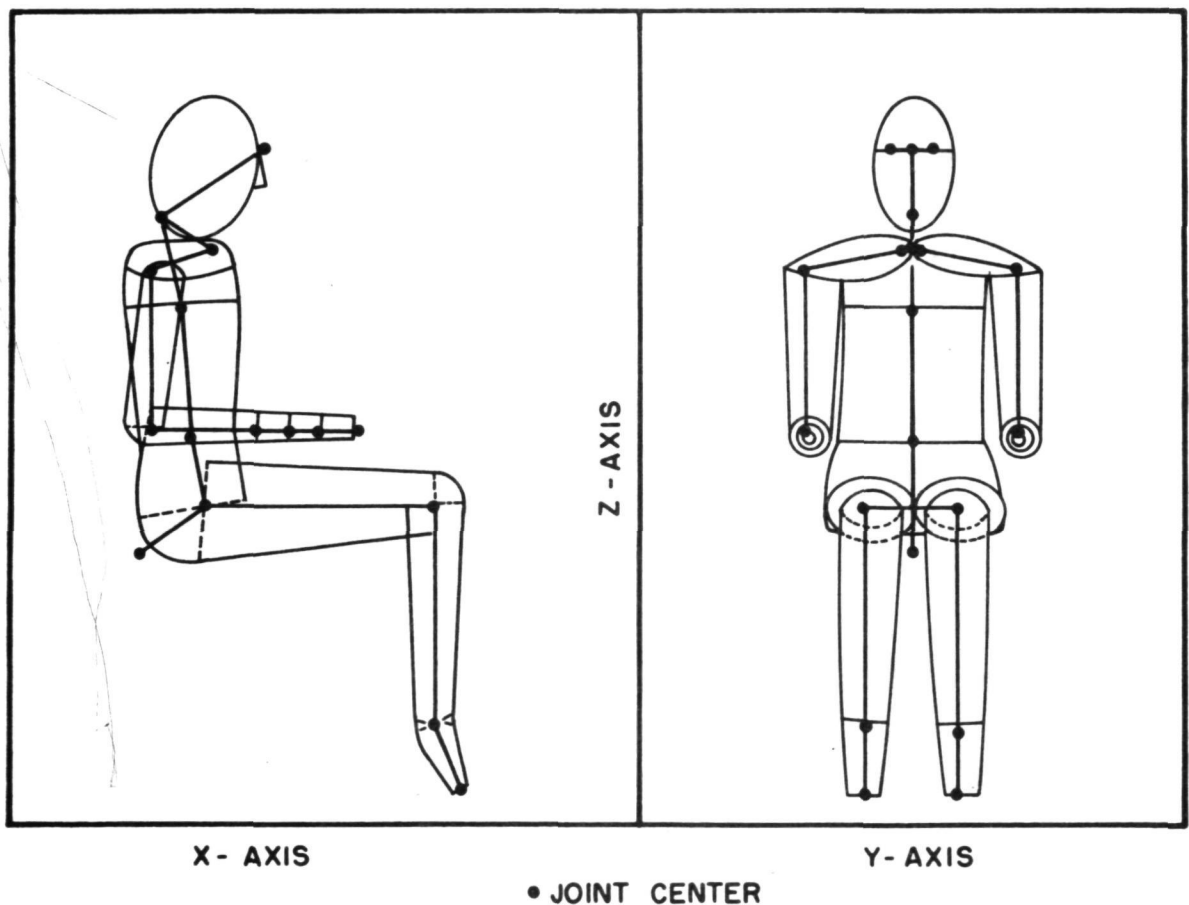


Figure 4.- Enfleshed COMBIMAN man-model.

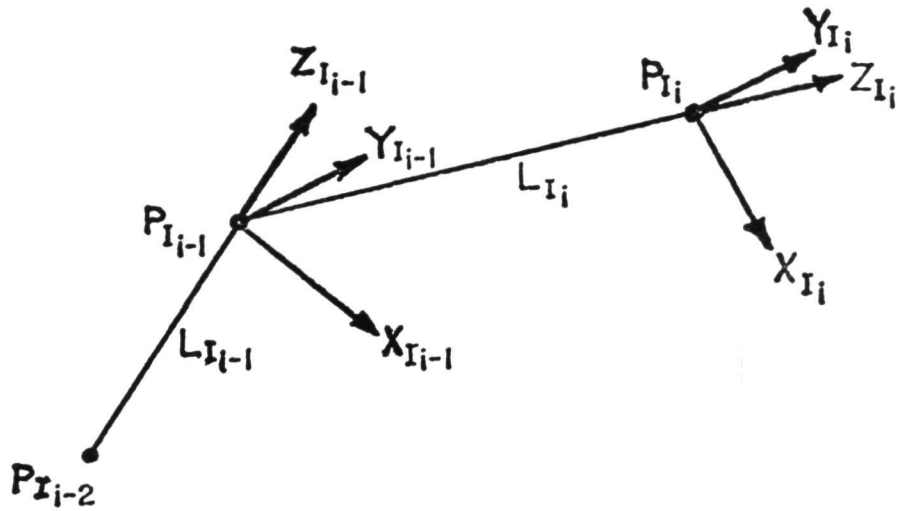


Figure 5.- Typical local coordinate systems affixed to links.

$$T_{I_i} = \begin{bmatrix} \cos \theta_{I_i} \cos \phi_{I_i} \cos \psi_{I_i} - \sin \phi_{I_i} \sin \psi_{I_i} & -\cos \theta_{I_i} \cos \phi_{I_i} \sin \psi_{I_i} - \sin \phi_{I_i} \cos \psi_{I_i} & +\sin \theta_{I_i} \cos \phi_{I_i} \\ \cos \theta_{I_i} \sin \phi_{I_i} \cos \psi_{I_i} + \cos \phi_{I_i} \sin \psi_{I_i} & -\cos \theta_{I_i} \sin \phi_{I_i} \sin \psi_{I_i} + \cos \phi_{I_i} \cos \psi_{I_i} & \sin \theta_{I_i} \sin \phi_{I_i} \\ -\sin \theta_{I_i} \cos \psi_{I_i} & \sin \theta_{I_i} \sin \psi_{I_i} & \cos \theta_{I_i} \end{bmatrix}$$

Figure 6.- Transformation matrix  $T_{I_i}$  for coordinate system  $C_{I_i}$ .

**Page Intentionally Left Blank**

**Page Intentionally Left Blank**

**Page Intentionally Left Blank**

AN IMPROVED HUMAN DISPLAY MODEL  
FOR  
OCCUPANT CRASH SIMULATION PROGRAMS

K. D. Willmert  
Mechanical and Industrial Engineering Department  
Clarkson College of Technology  
Potsdam, New York

and

T. E. Potter<sup>1</sup>  
Xerox Corporation  
Webster, New York

ABSTRACT

Presented is an improved three-dimensional display model of a human being which can be used to display the results of three-dimensional simulation programs that predict the positions of an occupant during impact of a vehicle. The model allows the user to view the occupant from any orientation in any position during the crash. The display model assumes the usual break up of the body into rigid segments which is normal for occupant crash simulation programs, but the shape of the segments in the display model are not necessarily the same as those used in the crash simulation. The display model is proportioned so as to produce a realistic drawing of the human body in any position. Joints connecting the segments are also drawn to improve realism.

---

<sup>1</sup>Formerly Graduate Assistant in the Mechanical and Industrial Engineering Department at Clarkson College of Technology, Potsdam, New York.



## INTRODUCTION

In simulating the motion of a human occupant during the crash of a vehicle, the normal approach is to represent the body as a collection of rigid members connected by appropriate joints. The positions of the human at successive increments of time are determined based on the dynamics of the assemblage of members taking into account their mass and moments of inertia, joint properties, etc. There is generally little regard, in these simulation models, to the actual shape of the body parts. After the positions of the segments have been determined, the body could easily be drawn on an x - y plotter or graphic computer terminal in stick form or using some other simple display representation of the body parts. The approach taken by Calspan Corporation [1] in their three-dimensional simulation program was to represent each segment in the display model as an ellipsoid. Two views of the outline (or shadowlines) of these ellipsoids were then drawn as shown in Figure 1.

In many applications, it is desirable to have a more realistic model of the human body that can be used to display the results of these occupant simulation programs. This was the goal of the work presented in this paper. It was assumed that the coordinates of the centers of gravity and direction cosine matrices (or Euler angles) of each body segment at all positions to be displayed were available as output from the simulation programs. Presented here is a means of taking this information and constructing a realistic body around the individual segments, connecting them with appropriate joints and displaying the resulting model on an inexpensive two-dimensional graphic computer terminal.

## DISPLAY MODEL

The body segments of the display model developed in this work are represented by non-uniform elliptic cylinders as shown in Figures 2 and 3. The surface of these cylinders can be expressed mathematically in terms of the  $X''$ ,  $Y''$ , and  $Z''$  coordinate system whose origin is at some reference point within the segment, generally the center of gravity, as shown in Figure 2. As functions of parametric coordinates  $s$ ,  $t_1$ , and  $t_2$  this surface is defined as:

$$X''(s, t_1, t_2) = (1-s)(A_1 \cos t_1 + C_1) + s(A_2 \cos t_2 + C_2) \quad (1)$$

$$Y''(s, t_1, t_2) = (1-s)B_1 \sin t_1 + s B_2 \sin t_2 \quad (2)$$

$$Z''(s, t_1, t_2) = (1-s)L_1 + s L_2 \quad (3)$$

where:

$A_1$ ,  $B_1$ ,  $A_2$ , and  $B_2$  are the ellipse semi-axis dimensions as shown in Figure 3.

$C_1$  and  $C_2$  are the offsets from the  $X''$ -axis at the corresponding ellipse end.

$L_1$  and  $L_2$  are the  $Z''$ -displacements of each end ellipse from the center of gravity (the origin of the segment coordinate system).

$M_1$  and  $M_2$  are constants used to change the length of the outside surface relative to the location of the end ellipses.

$s$  is the axial parametric coordinate.

and

$t_1$  and  $t_2$  are the circumferential parametric coordinates for the end ellipses.

The parametric coordinates are limited to the following ranges:

$$M_1 \leq s \leq 1 + M_2 \quad (4)$$

$$0 \leq t_1 \leq 2\pi \quad (5)$$

$$0 \leq t_2 \leq 2\pi \quad (6)$$

In order that Equations (1), (2), and (3) define a surface there must be a relationship between  $t_1$  and  $t_2$ . This is (see reference [2] for details):

$$A_2 B_1 \tan t_2 = A_1 B_2 \tan t_1 \quad (7)$$

By adjusting the quantities  $A_1$ ,  $B_1$ , etc., each segment of the human body can be represented fairly realistically.

The surface in a coordinate system  $X$ ,  $Y$ ,  $Z$  attached to the screen of the graphic terminal, where  $Z$  is perpendicular to the screen, can be obtained from:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \bar{A} \bar{D} \begin{bmatrix} X'' \\ Y'' \\ Z'' \end{bmatrix} + \bar{A} \begin{bmatrix} X'_{CG} \\ Y'_{CG} \\ Z'_{CG} \end{bmatrix} + \begin{bmatrix} X_T \\ Y_T \\ Z_T \end{bmatrix} \quad (8)$$

where  $X''$ ,  $Y''$ , and  $Z''$  are defined in Equations (1), (2), (3),  $\bar{D}$  is the direction cosine matrix of the segment,  $X'_{CG}$ ,  $Y'_{CG}$ ,  $Z'_{CG}$  is the position of the center of gravity of the segment,  $\bar{A}$  is the direction cosine matrix associated with the users desired viewing direction of the body, and  $X_T$ ,  $Y_T$ ,  $Z_T$  correspond to displacements needed to assure that the body appears approximately at the center

of the terminal screen. It is assumed that  $\bar{D}$ ,  $X'_{CG}$ ,  $Y'_{CG}$ , and  $Z'_{CG}$  are available as output from the occupant simulation programs. The yaw, pitch and roll angles which determine the  $\bar{A}$  matrix are selected by the user to give any desired view of the body.

The shadow lines (or outlines) of the body segments in the screen coordinate system can be determined by noting that the Z component of the normal to the surface along these outlines must be zero. Thus:

$$n_z = \frac{\partial X}{\partial t} \frac{\partial Y}{\partial s} - \frac{\partial Y}{\partial t} \frac{\partial X}{\partial s} = 0 \quad (9)$$

where  $n_z$  is the Z component of the normal. As  $s$  varies from 0 to 1, Equation (9) gives the value of the  $t$  parameter for the shadow lines. Since these are straight lines, only the end points need be determined. Evaluating (9) at  $s = 0$ , and also  $t = t_1$ , results in:

$$\left( \frac{\partial X}{\partial t_1} \frac{\partial Y}{\partial s} - \frac{\partial Y}{\partial t_1} \frac{\partial X}{\partial s} \right) \Big|_{s=0} = 0 \quad (10)$$

Using Equation (7) to eliminate  $t_2$  from this expression results in a fourth degree polynomial in  $\cos t_1$  which can be solved explicitly for values of  $t_1$  producing the shadow lines. The X, Y, Z coordinates of one end of the shadow lines can then be determined using Equations (8). The other end can be determined by using Equation (7) to find the  $t_2$  values corresponding to  $t_1$  obtained by solving Equation (10). Further details on this procedure can be found in References [2], [3], [4]. A display of the resulting shadow lines for the entire body are shown in Figure 4.

In order to complete the display, realistic joints were added between the segments. This was accomplished using circular arcs, straight lines, and clip and crease connections. Details of these connections can be found in References [2], [3], [4], and [5]. The final display is shown in Figure 5. Using different yaw, pitch and roll angles (selected by the user), other views of the body can be obtained as shown in Figure 6. Figures 5 and 6 show the same position of the occupant as that in Figure 1 produced by Calspan. As can be seen more realistic diagrams can be obtained from this improved display model.

Recently several modifications and improvements have been made in the basic display model. First, a change has been made in the approach used to decide which of the two shadow lines between adjacent segments must be connected. In the original approach the shadow lines to be connected were selected on the basis of the  $t$  parameter in the definition of the surface of the elliptic cylinder. The shadow lines of two adjacent body segments corresponding to the most similar  $t$  parameters at the appropriate ends were connected. This gave acceptable results in most situations. However at times the wrong shadow lines were connected, as shown in Figure 7. The new approach is based

on the cross product of two imaginary vectors in each segment as shown in Figure 8. The shadow lines are  $(X_1, Y_1)$  to  $(X_2, Y_2)$  and  $(X_3, Y_3)$  to  $(X_4, Y_4)$  for one segment and  $(X_5, Y_5)$  to  $(X_6, Y_6)$  and  $(X_7, Y_7)$  to  $(X_8, Y_8)$  for the other. The imaginary vectors are defined as:

$$\begin{aligned}\vec{V}_1 &= (X_2 - X_3, Y_2 - Y_3) \\ \vec{V}_2 &= (X_4 - X_1, Y_4 - Y_1) \\ \vec{V}_3 &= (X_6 - X_7, Y_6 - Y_7) \\ \vec{V}_4 &= (X_8 - X_5, Y_8 - Y_5)\end{aligned}\tag{11}$$

If the sign of the cross product  $\vec{V}_1 \times \vec{V}_2$  is opposite from that of  $\vec{V}_3 \times \vec{V}_4$ , then a connection is drawn from  $(X_2, Y_2)$  to  $(X_6, Y_6)$  and from  $(X_4, Y_4)$  to  $(X_8, Y_8)$ . Otherwise  $(X_2, Y_2)$  connects with  $(X_8, Y_8)$  and  $(X_4, Y_4)$  connects with  $(X_6, Y_6)$ . This approach seems to produce the proper connections. Figure 9 shows the same view as in Figure 7 except the determination of which shadow lines are to be connected is accomplished using this new method.

The original display program was written in BASIC on a small minicomputer. We have recently converted the program to FORTRAN on a commercial time sharing system, and have been investigating the display of output from other occupant simulation programs. Figures 10 and 11 show two positions taken from the UCIN program developed at the University of Cincinnati [6]. In the UCIN model only 12 body segments were used, as compared to 15 for the Calspan model. The feet and neck were not included as separate segments in the UCIN simulation, but were lumped with the other segments of the body. However in displaying the body, feet and neck certainly add realism, and thus data was created for these segments from the 12 existing ones to use in the plot.

#### EXTENSIONS UNDER DEVELOPMENT

Several additional modifications of the display model are currently under development. The approach used to connect segments to form realistic joints works well for the head-neck, upper-lower leg, upper-lower arm and between the three torso segments for all position and viewing angles. Likewise the connections between the neck and upper torso and between the lower torso and upper legs appear realistic in most cases as shown in Figures 5 and 11; however, for some positions of the body and viewing angles these two connections could be improved. Figure 6 shows a left shoulder which is somewhat distorted. The arm appears higher than it should be because of the connection which is made between the neck and upper torso. It thus seems that at times the neck should be connected to the arms rather than upper torso to gain better realism. Figure 10 shows an extra curve between the upper leg and lower torso in the side view and a distorted buttocks in the front view.

Modifications are currently underway to improve these two connections in order to obtain a better display model. Also under investigation are techniques to remove hidden lines from the display to further improve realism.

#### ACKNOWLEDGMENT

The authors would like to express their appreciation to the Office of Naval Research for their support of this research at Clarkson College of Technology through Contract No. N00014-70-A-0311-0003.

#### REFERENCES

1. Fleck, J. T., Butler, F.E., and Vogel, S. L., "An Improved Three Dimensional Computer Simulation of Motor Vehicle Crash Victims," Report No. ZQ-5180-L-1, Calspan Corporation, Buffalo, New York, July 1974.
2. Potter, T. E., "Three Dimensional Human Display Model for Two Dimensional Computer Graphics," Report No. MIE-006, Department of Mechanical and Industrial Engineering, Clarkson College of Technology, Potsdam, New York, June 1975.
3. Potter, T. E. and Willmert, K. D., "Three-Dimensional Human Display Model," Proceedings of the Second Annual Conference on Computer Graphics and Interactive Techniques, Bowling Green State University, June 25-27, 1975, pp. 102-110.
4. Potter, T. E., and Willmert, K. D., "Three-Dimensional Human Display Model," Report No. MIE-010, Department of Mechanical and Industrial Engineering, Clarkson College of Technology, Potsdam, New York, July 1975.
5. Willmert, K. D., "Occupant Model for Human Motion," Journal of Computers and Graphics, Vol. 1, 1975, pp. 123-128.
6. Passerello, C. E., and Huston, R. L., "User's Manual for UCIN Vehicle-Occupant Crash-Study Model," Report No. ONR-UC-EA-050174-2, University of Cincinnati, Cincinnati, Ohio, May 1974.

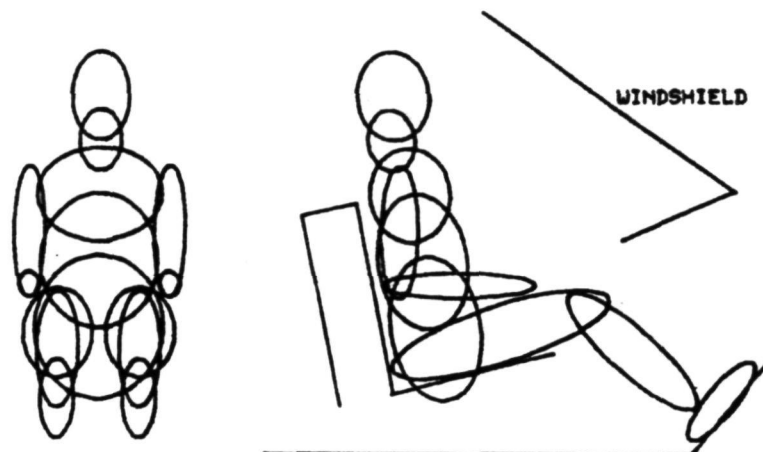


Figure 1.- Calspan display model.

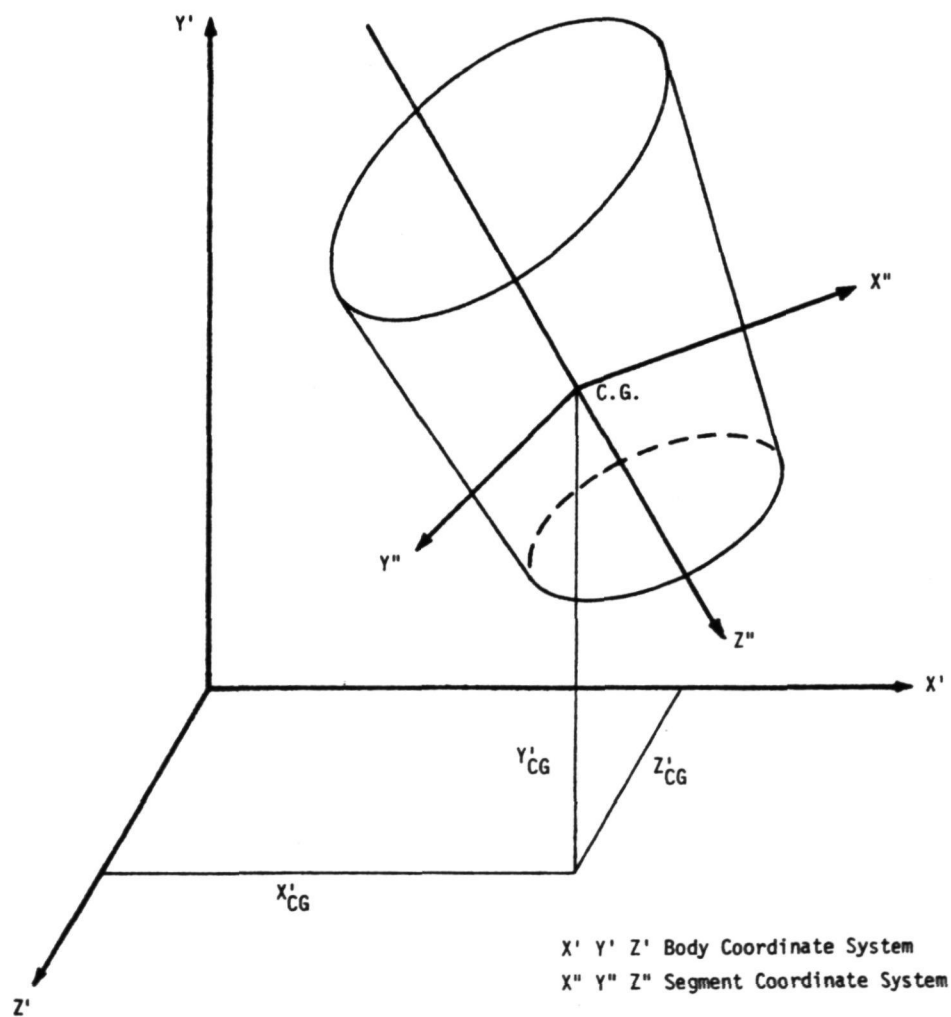


Figure 2.- Non-uniform elliptic cylinder.

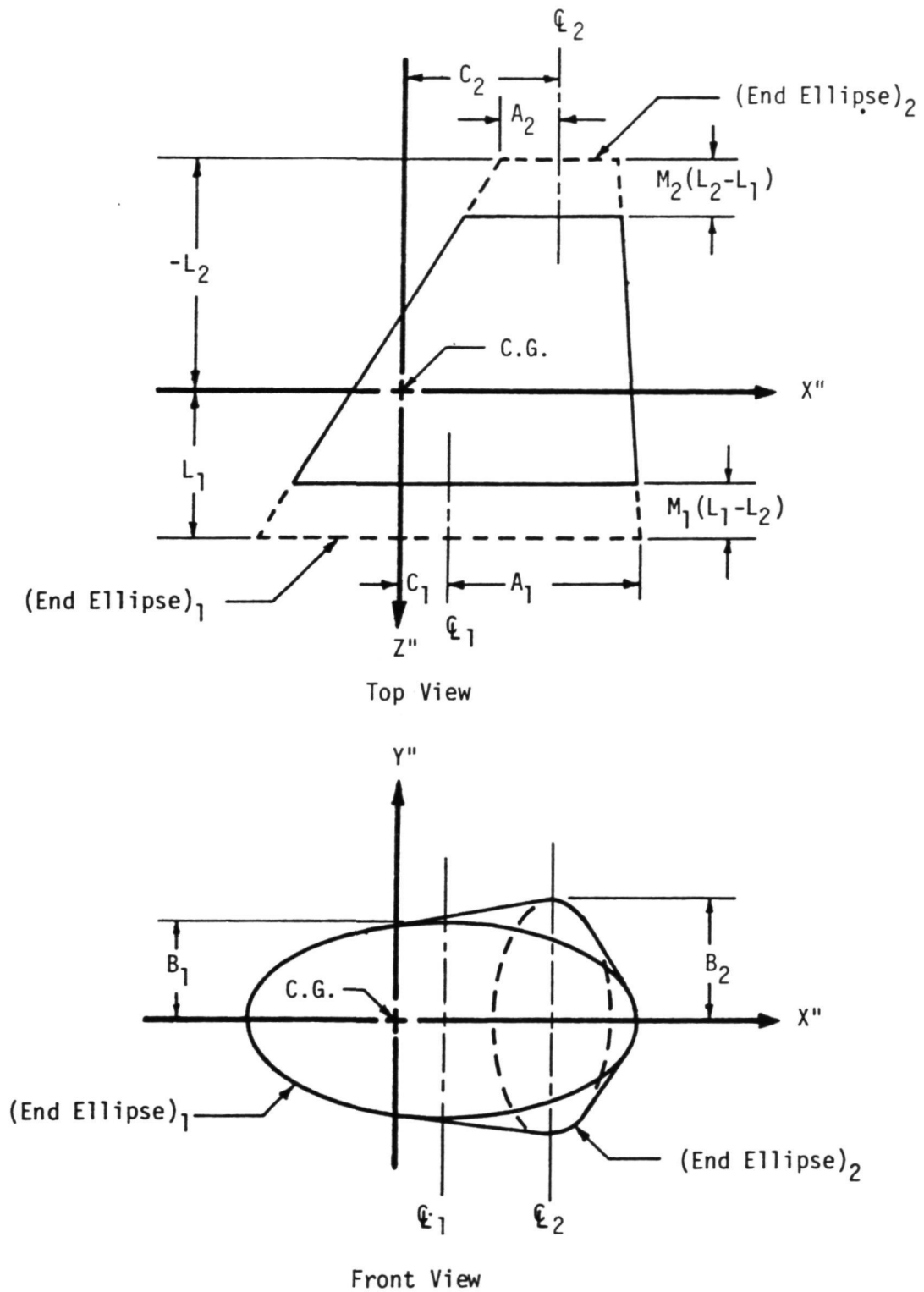
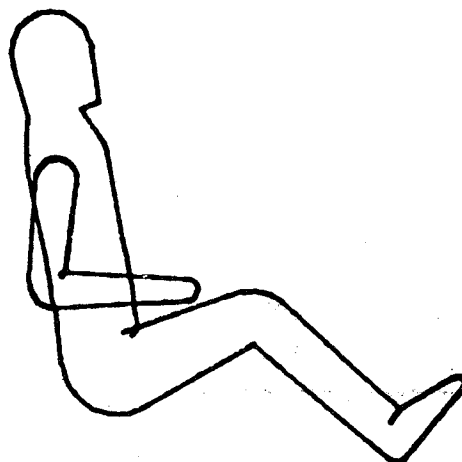


Figure 3.- Dimensions for non-uniform elliptic cylinder.



YAW • 270      PITCH • 0      ROLL • 0

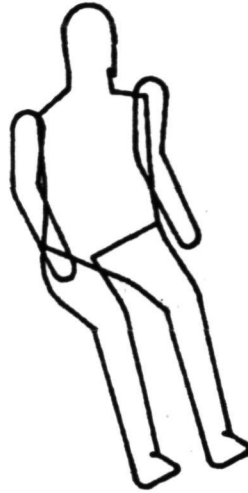
Figure 4.- Shadow lines.



YAW • 270      PITCH • 0      ROLL • 0

Figure 5.- Display with connections.





YAW = 200      PITCH = 45      ROLL = 0

Figure 6.- General orientation of the display model.

POSITION NUMBER DESIRED ?



YAW= 200.00    PITCH= 10.00    ROLL= 0.

Figure 7.- Display using old method of determining which shadow lines connect.

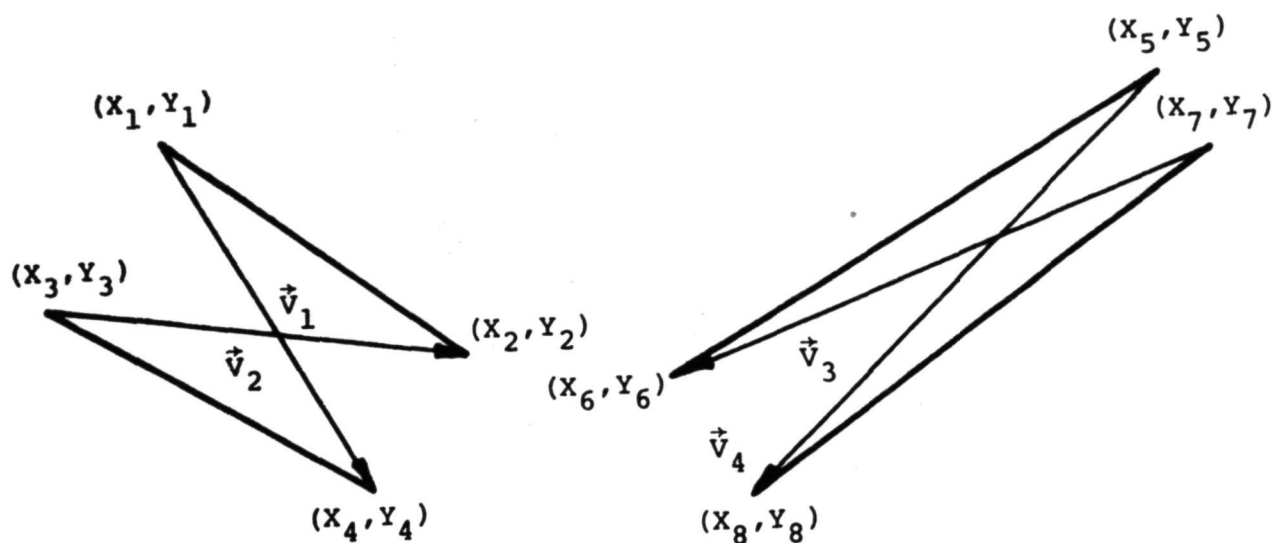
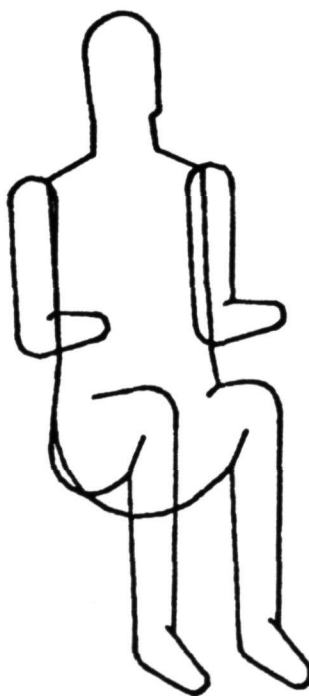


Figure 8.- New method of determining which shadow lines connect.

POSITION NUMBER DESIRED ?



YAW= 200.00 PITCH= 10.00 ROLL= 0.

Figure 9.- Display using new method of determining which shadow lines connect.

POSITION NUMBER DESIRED ?

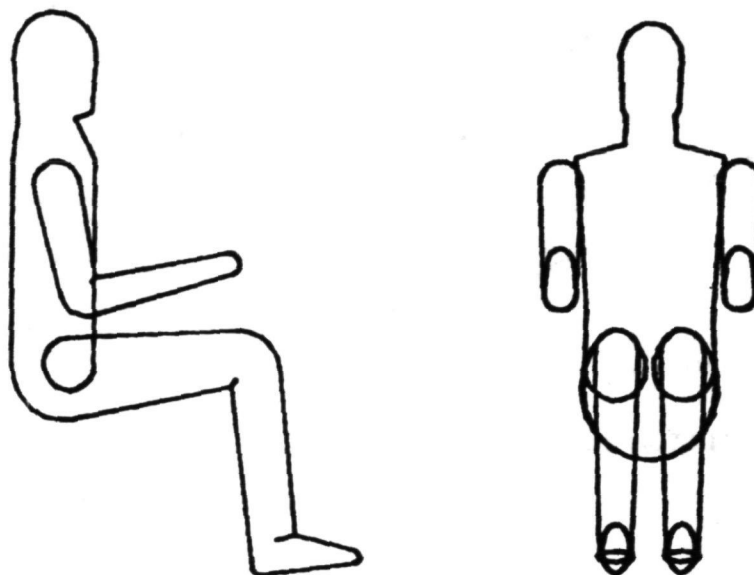


Figure 10.- First position from UCIN program.

POSITION NUMBER DESIRED ?

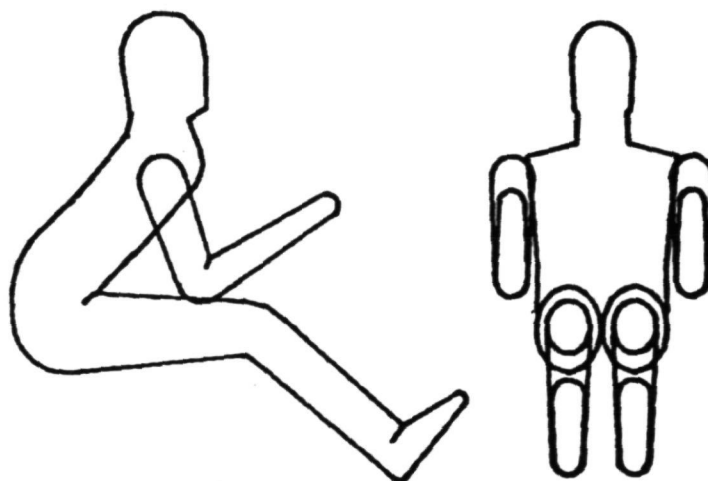


Figure 11.- Last position from UCIN program.

A FLEXIBLE FLIGHT DISPLAY RESEARCH SYSTEM  
USING A GROUND-BASED INTERACTIVE GRAPHICS TERMINAL

Jack J. Hatfield, Henry C. Elkins,  
Vernon M. Batson, and William L. Poole

NASA Langley Research Center

SUMMARY

Requirements and research areas for the air transportation system of the 1980 to 1990's are reviewed briefly to establish the need for a flexible flight display generation research tool. Specific display capabilities required by aeronautical researchers are listed and a conceptual system for providing these capabilities is described. The conceptual system uses a ground-based interactive graphics terminal driven by real-time radar and telemetry data to generate dynamic, experimental flight displays. These displays are scan converted to television format, processed, and transmitted to the cockpits of evaluation aircraft. The attendant advantages of a Flight Display Research System (FDRS) designed to employ this concept are presented. The detailed implementation of an FDRS, under development at Langley Research Center (LaRC), is described. The basic characteristics of the interactive graphics terminal and supporting display electronic subsystems are presented and the resulting system capability is summarized. Finally, the system status and utilization are reviewed.

INTRODUCTION

Aircraft in the air transportation system of the 1980 to 1990's must be capable of operating in a wide range of weather conditions and in congested airspace. These aircraft will be navigated and controlled with a greater precision, both in position and time, than is presently required. Advanced electronics technology is becoming available for implementation of avionics meeting these requirements. For example, advancements have been made in the areas of computer-generated electronic displays, computers for navigation and control, pilot input-output devices, and data link transceivers. Before this technology can be applied, however, much research is required to define the characteristics of a fully integrated system and the pilot's role as controller and system's manager. (See ref. 1.) The research must define the required pilot-vehicle interface, piloting procedures, avionic systems capabilities, and pilot-system task allocations.

An informal survey of personnel involved in vertical take-off and landing (VTOL) and fixed-wing aeronautical display research at LaRC indicated the need for a flexible display generation system to support flight projects in these areas. To provide the desired cockpit display research support, the following

capabilities are required:

1. Rapid development of displays, with minimal turn-around time for format modifications
2. Efficient generation of a wide spectrum of displays, from basic formats, such as conventional electro-mechanical indicators, through advanced formats, such as aeronautical chart and electronic attitude director indicator (EADI) displays, including true-perspective landing scenes
3. Simultaneous generation of multiple, independent displays
4. Generation of interactive, multi-mode displays, with automatic or pilot-initiated data entry and mode selection
5. Utilization of monochrome and color electronic display media
6. Mixing of display symbology with background scenes, such as those produced by airborne imaging sensors
7. Real-time generation of dynamic displays, with update rates from 0 to 30 hertz and refresh rates of 40 to 60 hertz.

A Flight Display Research System (FDRS) with these characteristics is being developed at LaRC to support both fixed-wing and VTOL aircraft display research. When its development is complete, the FDRS will enable researchers to have flight-test integrated, real-world pictorial, and symbolic displays which have the potential to declutter the cockpit and minimize the pilot's assimilation time. (See, for example, refs. 2 and 3.) In the following sections, the design concept for and the implementation of the FDRS are described and the resulting system capability is demonstrated. Finally, the developmental status, results of system testing, and utilization of the system are reviewed.

## DISPLAY EVALUATION SYSTEM

### Concept

Through an analysis of the preceding requirements and a survey of available computer-generated display systems, it was determined that an FDRS, using a ground-based interactive graphics terminal (IGT), could be developed to provide the required display generation capability. An IGT was chosen as the major FDRS subsystem for several reasons. In particular, interactive graphics display systems were judged to be more flexible, more easily programed (using higher order language), more capable of producing multiple, complex displays, and more readily accessible to interactive devices than were special-purpose, airborne, computer-generated display systems. The increasing use of interactive graphics display systems for ground-based cockpit simulator display

research supports this conclusion. (See, for example, refs. 4 and 5.)

A flight display testing method using the concept of ground-based display generation by an IGT is illustrated in figure 1. To evaluate a display using this concept, the evaluation aircraft would fly in the environment of the Wallops Flight Center (WFC) Experimental Runway Facility which would provide aircraft tracking, telemetry, and communications capability from an aeronautical radar research complex (ARRC). Prior to flight testing, the experimenters would prepare applications programs for generation of candidate display formats. An IGT, located within the FDRS, would provide the basis for rapid development and modification of these applications programs. During flight testing operations, real-time dynamics would be imparted to candidate flight displays by position, velocity, and simulated guidance data obtained from the ARRC and by aircraft sensor data telemetered from the evaluation aircraft via digital data link. The ground-generated flight displays would be converted into TV format, processed, and transmitted by video link to the evaluation aircraft where they would be displayed on high-performance TV monitors. Television was chosen as the method for remotng the visual capability of the IGT into the research cockpit because of many potential advantages. Processing, recording, transmitting, and displaying visual data in television format would result in standardized components which are less expensive than graphics format devices. Converting graphics displays to television format would permit simplified mixing of symbology with background scene generators which also use television formats. Processing images using sophisticated television video techniques would offer the potential for generation of advanced-format displays. Pilot interaction with the ground-based IGT would be provided functionally via the extensive up and down digital telemetry link and visually by the video uplink which closes the real-time in-flight simulation loop. The evaluation aircraft would also be capable of interacting with a simulated air traffic control (ATC) system via a Wallops and Langley data link.

Feasibility of the ground-generated, televised display concept was proven through prior flight research on graphic displays for steep, noise-abating approaches to landing (ref. 6). The display generation in this prior effort was provided by an analog computer and the displays investigated were necessarily very simple. Implementation of the concept using an IGT and correspondingly more sophisticated television processing techniques provides a method for satisfying the requirements of advanced aeronautical display research. This method has the following attendant advantages:

1. Generating the displays on the ground will permit the use of laboratory-quality equipment within the FDRS and simple airborne equipment within the evaluation aircraft.
2. Using an IGT in the ground-based configuration will make computer-generated display capability available to many aircraft research projects.
3. Basing the FDRS capability around an IGT will permit automation of pre-flight system checkout and post-flight data processing of quick-look performance data.

4. Matching the characteristics of the IGT used at WFC for flight research to those used at LaRC for research simulation support will provide software compatibility to smooth the transition from ground-based to in-flight display evaluation.

These are potential advantages which cannot be achieved without a detailed system design and development of components with appropriate performance characteristics. The characteristics of the IGT, the television subsystem, and the data telemetry subsystem are particularly important. In addition, extensive systems integration and interfacing are required. The following section describes the FDRS which resulted from such a detailed design, development, and systems integration process.

### System Design

The design and development challenge presented by the preceding display evaluation concept is to visually and functionally remote the capability of an IGT and its supporting display electronics, operating in real time, from the ground into the research aircraft. The design approach selected to accomplish these objectives is illustrated schematically in figure 2. Three basic communications paths are required between the ground-based equipment and the research aircraft. These paths are the radar tracking, the uplink and downlink data telemetry, and the television uplink.

The ARRC facility uses an FPS-16 radar, which can track aircraft accurately from 64.4 kilometers (45 miles) to touchdown, with telemetry coverage, and a laser tracker, which supplements the radar from 16.1 kilometers (10 miles) to touchdown. ARRC data processing is provided by two computers. A Honeywell 316 computer is used for processing radar/laser azimuth, elevation, and slant-range data to derive aircraft position, velocity, and simulated Instrument Landing System (ILS) guidance signals relative to user-selected runways and/or touchdown pads. A Honeywell 716 computer is used for management of digital communications between the Honeywell 316 computer, the LaRC/Wallops data link, the uplink and downlink telemetry, and the FDRS. The Honeywell 716 can also be utilized to derive simulated Microwave Landing System (MLS) guidance signals.

The telemetry system provides a means for transmitting both aircraft sensor outputs and pilot interactive commands to the ground. It also provides a means for transmitting computed data and ATC commands from the ground to the aircraft. This uplink and downlink telemetry system, known as the Transponder Data System (TDS), was developed specifically to LaRC and WFC specifications to support the FDRS. Data accuracy, capacity, and rate characteristics for the TDS were specified in response to requirements for aircraft data input and output to and from the FDRS. For example, it was specified that the TDS contain both proportional and discrete (digital on/off status) telemetry channel capacity to provide data for real-time animation of graphics displays and pilot interaction via airborne switch array and/or keyboard. The TDS operates through the FPS-16 transmitter/receiver and the aircraft transponder using the dead-time between ranging pulses to provide a

cost-effective pulse-position-modulated (PPM) digital data link. A summary of the TDS characteristics is contained in table I.

Figure 3 illustrates the ARRC facility's radar tracking, data processing, and telemetry subsystems in greater detail than figure 2. In addition, the general configuration, major components, and signal flow from and to the FDRS are shown in greater detail. Illustrated are extensive digital, analog, and discrete signal interfaces between the ARRC facility's subsystems, the IGT, and the hybrid interface and patching subsystem within the FDRS. These interfaces support the flow of radar-derived, telemetry, and LaRC/Wallops data link signals through primary and backup data paths.

Primary data flow between the ARRC and the FDRS is via digital data path between the Honeywell 716 and the IGT. This high-speed parallel digital interface is capable of 60 000 16-bit word transfers per second in a half-duplex mode. All transfers are initiated by the Honeywell 716. The IGT interface includes hardware packing and unpacking of two 16-bit Honeywell 716 words into one 30-bit IGT word.

Backup data flow between the ARRC and the FDRS is via analog and discrete data paths between the Honeywell 316 computer, the TDS, and the hybrid interface and patching subsystem of the FDRS. In the backup operational mode, high-accuracy proportional data (16 to 26 bits, binary) will be encoded in a coarse/fine two-channel analog data format, whereas standard-accuracy proportional data (10 to 13 bits, binary) will be encoded in a single-channel analog data format. This backup operational mode will permit many flight operations to continue in the event of failure of digital interfaces between the HW 316 and HW 716, the HW 716 and TDS, and/or the HW 716 and the IGT. The formats and the interface requirements for input and output of data to and from the IGT and its hybrid interface and patching subsystem are summarized by table II.

Figure 3 delineates and illustrates the two major subsystems of the FDRS in greater detail than in figure 2. These two subsystems are the interactive graphics terminal and the supporting display electronics. These subsystems and their associated interfaces are contained in the mobile instrument van shown in figure 4. Developing the FDRS in a van permits its use at WFC for flight research support and at LaRC for simulator research support. The van is equipped with an air-ride suspension system, equipment shock mounting, and an air conditioning system for appropriate environmental control of the IGT and the supporting display electronic subsystems.

The IGT subsystem forms the basis for:

1. Programing and modifying advanced flight display formats
2. Processing radar-derived and telemetry data inputs for real-time display animation



### 3. Generation of the dynamic graphics displays

4. Automating control of static and dynamic background image generators represented in figures 2 and 3.

The static and dynamic background images as well as the dynamic graphic images are converted into television format as represented by figures 2 and 3. Once these images are in television format, they can be easily processed and mixed using television studio equipment. This processing and mixing can be designed to achieve advanced-format display generation capability, such as grey-scale or color encoding of symbology and prioritizing or windowing of symbology.

The processed television video can be recorded and transmitted to the aircraft via a microwave uplink which has dual, independent, high-resolution channels. This television uplink has the capacity to handle both monochrome and color-encoded video. Resolution capability for each independent channel is variable from a low of 350 x 300 TV lines for National Television Standards Committee (NTSC) color-encoded channels (at a scanning standard of 525 TV lines per frame) to 850 x 1,000 TV lines for monochrome channels (at a scanning standard of 1225 TV lines per frame). The projected transmission range for the television link is 32.2 kilometers (20 miles). Display of the dual independent video channels in the research aircraft is via high-performance monochrome or color television monitors. When used to support ground-based cockpit simulation at LaRC, the television uplink of the FDRS need not be used, since the video output channels from the TV processing and mixing electronics illustrated in figure 3 can be used to drive high-performance television monitors directly.

This section has emphasized the general systems configuration of the FDRS, the description of the ARRC facility, and signal flow between the ARRC, the FDRS, and the research aircraft. The capability of the FDRS within the configuration described is critically dependent on the characteristics of its two major subsystems. The salient features of these subsystems and their effect on system capabilities are described in the next section.

## FLIGHT DISPLAY RESEARCH SYSTEM

### Interactive Graphics Terminal

As related in the previous section on design approach, the FDRS is configured around the capabilities of the IGT, its associated peripheral devices and interfaces, and its graphics displays. These components are illustrated in simplified form in figure 3 and in detail in the IGT block diagram of figure 5. Figures 6(a) and 6(b) show the actual IGT in the FDRS mobile equipment van. The IGT is an Adage AGT/130 Graphics Terminal System. The major components are its DPR-4 digital processor, disk memory subsystems, digital/hybrid interfaces with associated transformation array, vector and character generators, interactive devices, and multiple graphics displays.

Location of the IGT at a remote site, such as the WFC Experimental Runway Facility, required that it have real-time, dynamic display generation capability without dependence on a large host computer. This capability is provided by the graphics-oriented processor in conjunction with the hybrid coordinate transformation array of figure 5. The 30-bit word length of the digital processor speeds up image manipulations by using one-word and two-word formats to define two-dimensional and three-dimensional vectors, respectively, in its outputs to the coordinate transformation array. This array has the capability to scale, translate, and rotate these vectors at a rate of 4 microseconds per vector. This speed is attained through utilization of hybrid electronics operating in parallel to solve the direction-cosine matrix. The coordinate transformation array drives graphics displays through the vector generator and character generator. The character generator is capable of producing alphanumeric elements in four sizes.

The digital processor has 16K words of storage and is programable in Fortran IV and/or assembly language. The processor and disk memory are software-compatible with computer graphics terminals presently being used at LaRC to support real-time cockpit display simulations (ref. 7). Thus, applications programs for flight display research, as well as for simulator display research, can be written in Fortran IV and minimal software modifications will be required to translate a display evaluation program from the cockpit simulator environment to the flight environment.

The disk memory subsystems provide a basis for storage of source and relocatable object programs as well as a monitor and operating system which automates programing and system control. The operating system permits use of the console display and keyboard/teletypewriter to manage disk files, edit, and compile and assemble programs, and to monitor and evaluate displays. Disk packs used on the 80M-bit disk memory of figures 5 and 6(b) are interchangeable with LaRC units. The second memory (40M-bit) of figures 5 and 6(b) provides a backup operational capability in the event of failure of the primary disk drive.

Interactive capability is provided at the IGT console by discrete interrupt function switches, alphanumeric keyboard, variable analog voltage control dials, analog data tablet, light pen, and a joy stick illustrated in figure 5 and shown in figure 6(a). Much of this interactive capability can be remoted to the aircraft cockpit via the digital, discrete, and analog interfaces illustrated in figures 3 and 5 and shown in figure 7. Figure 7 shows the portion of the hybrid interface and patching subsystem which is implemented outside of the IGT main frame. This subsystem comprises the FDRS input and output signal interfacing rack (utilizing differential buffering techniques on both analog and discrete signals), an analog computer (for special buffering and subsystem testing), and analog and discrete signal patch panels (for routing of signals to, from, and within the FDRS). In addition, an analog tape recorder is available for recording signals selected at the analog patch panel. The IGT drives or receives signals from all remote devices, either internal or external to the FDRS, through these interfaces. For example, they provide the path for IGT programatic control of FDRS image sources, such as the static image files and moving maps (backgrounds)

illustrated in figure 3. Since the characteristics of these interfaces are summarized in table II, they will not be repeated here.

The graphics displays available to the IGT user include the graphics console display and three precision graphics displays as illustrated in figure 5. Four independent displays can be created simultaneously and refresh and update rates are software selectable. This capability includes the programmatic segregation of symbology intended for a single composite display onto multiple graphics displays. In this manner, the scan-converted symbology from each graphics display can be subjected to individual video processing techniques. This capability is a key factor in attaining the required special effects for advanced-format display generation. Figure 6(a) shows a three-dimensional test pattern displayed on the graphics console display and that same test pattern displayed on the television monitor at the left, after having been scan converted to television format. The scan-conversion process is described in the next section on the supporting display electronics subsystem. A summary of the hardware and software characteristics of the IGT is contained in table III.

#### Supporting Display Electronics Subsystem

Graphics-to-TV scan conversion. - The graphics-to-TV scan-conversion technique used in the FDRS is illustrated in figures 2 and 3 and shown in figure 8(a). This latter figure shows two of the IGT precision graphics displays being viewed by two high-resolution television cameras and thereby forming electro-optical, graphics-to-TV scan-conversion channels. The graphics displays shown are precision, flat-faced, stroke-drawn displays having a minimum of 2000 resolvable picture elements (equivalent of 2000 TV scan lines) per display diameter. The clarity of the precision displays is illustrated by the photographic insets in figure 8(a) showing an EADI flight display on the left cathode ray tube (CRT) and an aeronautical chart flight display on the right CRT. This figure demonstrates the IGT capability to generate dual, independent, complex displays simultaneously. In the scan conversion process, the photoconductive surface of the vidicon imaging tube provides short-term storage of graphics images (for one TV frame), until erased by the TV scanning raster in the readout process. This method provides flicker-free scan conversion of the graphics images into television format (provided that the graphics images are refreshed at a 40- to 60-hertz rate). The television rendition of the graphics display loses some clarity and edge sharpness; however, much of this clarity and edge sharpness can be recovered by scanning the television cameras at a high line rate (from 875 to 1225 TV lines per frame) or by television image enhancement. The FDRS television subsystem has both of these capabilities.

Background scene generation. - Figure 3 illustrates the image sources controlled by the IGT. In addition to the graphics displays discussed above, the static and dynamic background scene generators shown are programatically controlled. These background scene generators will consist of future image sources, such as a rastergraphic display generator and a visual landing scene generator presently under development, static image files, and moving-map

image sources. The static image files and moving-map image sources, as well as television video recording components, are shown in figure 8(b). The static image files comprise two dual-drum projectors, operating under IGT programatic control and imaging into two high-resolution vidicon TV cameras. The moving-image sources (moving maps) comprise two back-lighted, motion-base transparency tables imaging through pechan (roll) prisms and zoom lenses, all of which are under IGT programatic control, into two high-resolution vidicon TV cameras. The video recording subsystem comprises two helical-scan tape recorders, which can record independent displays at variable scan standards, and a video hard copy unit which can record displays at 525 scan lines/frame.

Video processing and mixing techniques. - Since television processing and mixing technology must provide all required capabilities not inherent in the IGT, much effort has been devoted to implementing a flexible, high-performance processing and mixing subsystem. In particular, video processing techniques are key factors in generating advanced-concept displays having simulated rastergraphic, color-coded, and prioritized or windowed symbology. Flexibility is achieved through use of components which can operate at TV scanning standards from 525 lines/TV frame to 1225 lines/TV frame, and by video patching which can reconfigure subsystem architecture and signal flow. High performance is achieved through use of wide-bandwidth, temperature-stabilized components which produce high-resolution images for transmission.

Typical video processing which has been implemented within the FDRS for generation of advanced-format displays is illustrated in figures 9(a) and 9(b). The video processing of figure 9(a) produces two independent displays -- a simulated rastergraphic/stroke-drawn EADI on the upper high-resolution TV monitor and a stroke-drawn aeronautical chart display on the lower high-resolution TV monitor.

The video processing of figure 9(b) produces a single color-coded display -- a simulated EADI, which can be displayed as illustrated in figure 9(b) in red-green-blue (primary colors) format on the upper color TV monitor or in NTSC encoded format on the lower color TV monitor. The latter format is more suitable for transmission because both luminance and chromanance information are encoded onto one video channel, as opposed to the red-green-blue format which requires three video channels. The red-green-blue format is preferable for applications requiring high-resolution display because the NTSC encoding process limits the horizontal resolution.

As illustrated in figure 9(a), an EADI typically requires white symbology, black or grey-scale encoded symbology, prioritized or windowed symbology, and sky/ground shading. The processing technique for generation of a simulated rastergraphic/stroke-drawn EADI display is shown by figure 9(a) to require presentation of white symbology, black symbology (white symbology to be grey-scale encoded), and priority symbology on separate graphics scopes. In addition, image sources defining priority windows and sky/ground shading are presented on the FDRS background image generators. All image sources are converted into TV format. The nonadditive mixer combines the white symbology with sky/ground shading. Then the black symbology is matted into this composite image by using special effects generator number 1. The composite image

output is fed to special effects generator number 2 where the priority window symbology is inserted by using an external key mode, and thereby forming the final, composite image. The processing technique for generation of the simulated stroke-drawn aeronautical chart display is shown by figure 9(a) to require only scan conversion of the display, generated on one graphics scope, into TV format for presentation on a high-resolution TV monitor.

The processing technique of figure 9(b) produces multiple colors through treating the symbology or background image from each source as a different color. Therefore, the symbology on graphics scope 1 can be assigned color A, the symbology on graphics scope 2 can be assigned color B, etc. TV format renditions of the symbology on each graphics scope are processed by a switch and gain matrix. This matrix can assign any color to the symbology from a given scope through adjustment of the levels of three output signals from that scope into the red-green-blue, first-stage nonadditive mixer. Color coding is assigned to the sky/ground shading from the moving-map source of figure 9(b) by the same encoding process at the second-stage nonadditive video mixer. By using the described encoding process, the FDRS can generate four-color displays using the graphics scopes as image sources, and eight-color displays, using static and dynamic background image sources in addition to the graphics scopes.

#### System Capability

To demonstrate the advanced-concept display generation capability of the FDRS, a number of candidate simulator and/or flight displays have been programed in Fortran. For example, the EADI and aeronautical chart displays of figures 10(a) and 10(b) are important new concepts for integrated display of vertical and horizontal situation and command information in cruise and terminal phases of aircraft missions. They are presented here only to be illustrative of the complexity of simultaneous, dynamic displays which the FDRS is capable of generating and the video processing techniques which are used. Therefore, the display formats and their utilization by the pilot will not be described in detail. The simulated rastergraphic/stroke-drawn EADI display of figure 10(a) was generated by using the video processing techniques of figure 9(a). It is typical of formats being studied by researchers to present aircraft vertical situation and command information to pilots. It can contain from 10 to 20 symbolic and pictorial indications including an aircraft reference symbol, ILS deviation box, potential flight path symbol, artificial horizon, roll/pitch grid (simulated stroke-drawn, white symbology), sky/ground shading (simulated rastergraphic symbology), roll scale/pointer and flight director bars (prioritized, grey-scale encoded symbology), and alphanumeric presentations of altitude and navigational waypoints (priority-window symbology). The techniques of shading, grey-scale encoding and prioritizing of symbology provide a less cluttered and more easily interpreted display. For example, the use of sky/ground shading (ref. 1) has produced fewer control reversals by pilots.

The aeronautical chart display of figure 10(b) was generated by using the video processing techniques of figure 9(a). It is an example of a simulated stroke

display (all white symbology) employing no rastergraphic techniques. It is typical of formats being studied by researchers to present aircraft horizontal situation information to pilots. It can contain from 20 to over 100 symbolic and pictorial indications to present a plan-view map of radio navigation aids, airport symbols, range circles, holding patterns, and waypoints.

By using the interactive capability of the FDRS, evaluation pilots can add or delete symbology in the EADI display of figure 10(a) and change map scales or change from north-up to heading-up presentations in the aeronautical chart display of figure 10(b).

The EADI displays of figures 11(a) to 12(b) are presented to show the FDRS capability for color-coded display generation. Color coding of displays is of interest to aeronautical researchers as a means of decluttering displays and providing faster data identification. Reference 8, for example, has shown that color coding has an advantage over shape coding of symbology for target identification and counting.

Figures 11(a) and 11(b) provide a comparison of the monochrome presentation and the color-coded presentation of the EADI display. (Although these figures were presented in color at the conference, consideration of time and expense preclude their reproduction in color here). In these two displays, the sky/ground shading was produced by stroke-drawn vectors from the IGT. Figures 12(a) and 12(b) show the same EADI display with and without rastergraphic sky/ground shading produced by the processing technique of figure 9(b). The FDRS can change color coding either interactively or programatically. Thus, a color change can be provided to suit pilot preference or to signify an error or an alarm status indication. The color-coded displays of figures 11(b) and 12(a) are more suitable for presentation in a head-down cockpit display, whereas the color-coded display of figure 12(b) is more suitable for presentation in a head-up display (HUD) where the pilot must view an out-the-window scene through the symbology set.

The displays of figures 10(a) to 12(b) are illustrative of the broad-based display generation capability of the FDRS. These displays illustrate the clarity of a television format of 525 scan lines per television frame. Higher clarity can be achieved in the monochrome displays by using higher line rates, such as the 875 to 1225 scan lines per television frame, which the system has available.

### System Status

The FDRS, as described above, is in the final stages of development; however, the system as presently configured has broad-based cockpit display generation capability. In fact, the system has been used to support three research simulation projects at LaRC. To provide dynamic, real-time displays to both motion-base and fixed-base cockpits, the FDRS was interfaced to a CDC 6600 simulator computer by data lines and to the cockpits by video lines. The display formats produced by the FDRS for these simulations are indicative of the wide spectrum of display generation capability. The display formats are shown in figure 13 and are discussed briefly.

The top display is an EADI which was generated for LaRC's Terminal Configured Vehicles (TCV) project for display in the 737 motion-base simulator. Interactive capability was also remoted to the simulator to allow the pilot to change from cruise mode to landing mode and to add or delete symbology. The display shown is the landing mode which contains a perspective runway. The middle displays of figure 13 are a Helicopter Attitude Director Indicator and an Integrated Horizontal/Vertical Situation display which were generated for the LaRC VTOL Approach and Landing Technology (VALT) program for display in a CH-46 fixed-base simulator. The display on the left contains situation and command information and the display on the right contains situation and predictive information. The bottom display of figure 13 is a vertical scale, dial, and drift meter format which was generated for the VALT program for display in the SH-3A simulator. These conventional instrument renditions were video mixed with a color visual landing scene to provide situation information during helicopter approach, hover, and landing. All these user programs were written in Fortran.

To provide for the smooth integration of the FDRS with the Wallops facility, including the radar, the telemetry (TDS) and the research aircraft, operational testing was performed at WFC using a C-54 aircraft in early 1975. The status of the FDRS at the time of this testing was as follows:

1. All interfaces with ARRC were implemented with the exception of the digital interface between the interactive graphics terminal and the WFC HW 716. (See fig. 3.)
2. The video subsystem was used in an interim configuration which did not afford the full design capabilities.
3. The television uplink telemetry contained only a single channel capability as compared with the dual channel capability illustrated in figure 3.

The data flow concept, utilized for system checkout, was to loop uplink data telemetry, received on the test aircraft, back around into downlink data telemetry using all TDS uplink and downlink proportional and discrete channels. The interactive graphics terminal and television subsystem within the FDRS played key roles in that (1) the interactive graphics display system was the (a) source of telemetry test signals, (b) destination of looped-around telemetry test signals, (c) means of calculation and display of diagnostic presentations comparing uplink and downlink test signals, and (2) the television subsystem was the means for scan converting the diagnostic graphics displays and remoting them to ARRC and the test aircraft.

The operational testing at WFC verified anticipated data rates, delays, accuracy, transmission range, and imaging performance for the entire ARRC/FDRS/research aircraft system operating in a closed-loop, dynamic fashion. The tests demonstrated the feasibility of the ground-based interactive graphics terminal televised display concept, and its use in a diagnostic pre-flight and flight checkout mode. Examples of three diagnostic displays for presenting



and analyzing telemetry system performance are shown in figures 14(a) to 14(c), as reproduced on the FDRS television hard-copy unit.

In the diagnostic bargraph display of figure 14(a), a failed or an out-of-tolerance telemetry channel is readily identified. For example, note the simulated failure of the top nine downlink channels. In the diagnostic alphanumeric display of figure 14(b), values of uplink test signals and looped-around downlink signals can easily be compared and the IGT can calculate and display system loop-errors for each channel. This latter display mode is shown in figure 14(c).

Since the authors' return to LaRC from WFC, the planned development of the FDRS, primarily in the areas of the digital communications between the FDRS and the ARRC, the television processing subsystem, the television uplink system, and the operator's console, has been continued. In the spring of 1976, the completed system will begin operational support of TCV and VALT flight research projects at the WFC Experimental Runway Facility.

#### CONCLUDING REMARKS

A Flight Display Research System is being developed to support in-flight evaluation of proposed aeronautical displays. The system uses a ground-based interactive graphics terminal to achieve a general capability for producing complex displays. A telemetry data subsystem and a television processing and transmission subsystem are used to functionally and visually remote the capability of the graphics terminal from the ground into the research aircraft.

This technique has many advantages:

1. In-flight evaluation of cockpit display concepts can be accomplished without developing costly computer-based airborne systems. Generating the displays on the ground permits the use of laboratory quality equipment within the FDRS and simple airborne equipment within the evaluation aircraft.

2. Use of an IGT in the ground-based FDRS makes computer-generated display capability available to many aircraft research projects. This capability permits evaluation of a wide spectrum of displays -- from basic formats such as conventional electro-mechanical indicators, through advanced formats such as aeronautical charts and electronic attitude director indicator (EADI) displays, including true-perspective landing scenes. The graphics software required to develop these displays can be written in a high-level language to permit rapid program development and display format modification.

3. Matching the characteristics of the IGT used at WFC for flight research to those used at LaRC for research simulation support will provide software compatibility to smooth the transition from ground-based to in-flight display evaluation.



4. The processing and mixing of the images prior to transmission to the aircraft can be designed to enhance the inherent capability of the IGT in achieving advanced-format display generation capability. Specific capabilities afforded by this processing include color and grey-scale encoding of symbology, mixing symbology with rastergraphic or continuous tone scenes, prioritizing and windowing symbology, and providing a standard format for recording, transmission, and display.

The system design and component development effort required to achieve these advantages has been described. Display formats demonstrating system capability and the interim use of the system to support ground-based simulation have been presented. Operational testing at WFC has verified anticipated data rates, delays, accuracy, transmission range, and imaging performance for the entire ARRC/FDRS/research aircraft system operating in closed-loop, dynamic fashion. After this testing, system development has continued and the system is scheduled to commence support of flight research projects in the spring of 1976.

## REFERENCES

1. Wempe, T.: Flight Management - Pilot Procedures and System Interfaces for the 1980-1990's. AIAA Paper No. 74-1297, Nov. 1974.
2. Warner, John D.: Advanced Controls and Displays for Future Commercial Aircraft Operations. AIAA Paper No. 70-938, July 1970.
3. Mulley, William G.: Instrumentation Displays for Future Naval Aircraft. AIAA Paper No. 75-599, Apr. 1975.
4. Imrich, Thomas: Concept Development and Evaluation of Airborne Traffic Displays. FTL Rep. R71-2, Mass. Inst. Technol., June 1971.
5. Palmer, E. A.; and Cronn, F. W.: Touchdown Performance With a Computer Graphics Night Visual Attachment. AIAA Paper No. 73-927, Sept. 1973.
6. Elkins, Henry C.; and Hatfield, Jack J.: Televised Graphic Displays for Steep Approach-to-Landing Research. NASA paper presented at 11th National Symposium on Information Display (New York, N.Y.), May 1970.
7. Leavitt, John B.; Tarig, Syed I.; and Steinmetz, George G.: The Design and Implementation of CRT Displays in the TCV Real-Time Simulation. Applications of Computer Graphics in Engineering, NASA SP-390, 1975.
8. Smith, Sidney L., and Thomas, Donald W.: Color Versus Shape Coding in Information Displays. J. Appl. Psych., vol. 48, no. 3, June 1964.

TABLE I. - TRANSPONDER DATA SYSTEM CHARACTERISTICS SUMMARY

I. <u>GROUND-BASED TRANSMITTER/RECEIVER:</u>	C-BAND RADAR, RCA AN/FPS-16
II. <u>AIRBORNE TRANSMITTER/RECEIVER:</u>	C-BAND TRANSPONDER, VEGA MODEL 302C-2
III. <u>MODULATION TECHNIQUE:</u>	DIGITAL PPM: CODING INTERSPERSED BETWEEN RADAR/TRANSPONDER RANGE PULSES
IV. <u>TELEMETRY LINK DATA RATES:</u>	1.2 KILOBITS $\alpha$ 120 PRF 6.4 KILOBITS $\alpha$ 160 PRF 10.2 KILOBITS $\alpha$ 1024 PRF
V. <u>TYPE OF DATA TRANSMITTED:</u>	PROPORTIONAL AND DISCRETE CHANNELS
VI. <u>CHANNEL CAPACITY:</u>	SELECTABLE;
<u>UPLINK</u>	<u>DOWNLINK</u>
0 TO 16 PROPORTIONAL CHS. AND 0 TO 64 DISCRETE CHS.	0 TO 48 PROPORTIONAL CHS. AND 0 TO 64 DISCRETE CHS.
VII. <u>ENCODER CHANNEL SIGNAL CHARACTERISTICS:</u>	
<u>UPLINK</u>	<u>DOWNLINK</u>
PROPORTIONAL: $\pm$ 10 VOLTS ANALOG OR 10-BITS TTL PARALLEL DIGITAL MUXED, EACH CHANNEL SELECTABLE.	$\pm$ 5 VOLTS ANALOG OR 10-BITS TTL PARALLEL DIGITAL MIXED, SELECTABLE IN GROUPS OF 4 CHS.
DISCRETE: TTL INDIVIDUAL LINES OR TTL PARALLEL DIGITAL MUXED, SELECTABLE IN GROUPS OF 4 CHS.	TTL INDIVIDUAL LINES OR TTL PARALLEL DIGITAL MUXED, SELECTABLE IN GROUPS OF 4 CHS.
VIII. <u>ACCURACY AND RESOLUTION:</u>	
<u>ANALOG I/O</u>	<u>DIGITAL I/O</u>
ACCURACY: 0.5% OF FULL SCALE	0.3% OF FULL SCALE
RESOLUTION: 0.1% OF FULL SCALE	0.1% OF FULL SCALE
IX. <u>ERROR DETECTION CODES EMPLOYED:</u>	
PROPORTIONAL CHANNELS: 2-BIT PARITY CODE	
DISCRETE CHANNELS: VIT CODE	
ENTIRE FRAME: BCH CODE	
X. <u>SECURITY MODES:</u>	
MAXIMUM SECURITY: FRAMES WITH ERRORS IDENTIFIED BUT REJECTED	
MINIMUM SECURITY: FRAMES WITH ERRORS IDENTIFIED BUT ACCEPTED	
NORMAL SECURITY: INDIVIDUAL CHANNEL ERRORS IDENTIFIED BUT ACCEPTED OR REJECTED BY SWITCH SELECTION	
XI. <u>SPECIAL FEATURES:</u>	
PROPORTIONAL CHANNEL STRAPPING FOR INCREASED SAMPLING RATE (2 TO 8 CHS/STRAPPED CH.)	
PROPORTIONAL CHANNEL PAIRING FOR INCREASED ACCURACY (16- TO 20- BITS)	
HIGHER UPDATE RATES FOR SELECTED DISCRETES	

TABLE II. - INTERACTIVE GRAPHICS DISPLAY SYSTEM MAJOR INTERFACES

#	ROUTING		DATA TRANSMITTED	DATA TYPE	PATH TYPE	WFC	FDRS	CHANNEL CAPACITY
	FROM	TO						
(1)	HW 316 (D-to-A)	DPR4 (A-to-D)	A/C Position, Velocity, and Guidance Info.	ANALOG	BACKUP	±20 volts	±10 volts	10
(2)	DPR4 (D-to-A)	TDS Uplink Encoder	Computed Guidance and Control Info.	ANALOG	BACKUP	±10 volts	±10 volts	16
(3)	TDS Downlink Decoder	DPR4 (A-to-D)	A/C Air-Mass and Inertial Referenced Info.	ANALOG	BACKUP	±10 volts	±10 volts	48
(4)	DPR4 (D-to-A)	FDRS Television Subsystem	Control of Background Scene Dynamics	ANALOG	PRIMARY	NA	±10 volts	24
(5)	TDS Downlink Decoder	DPR4 (Discrete Interface)	Pilot Mode Selection and Data Entry	DISCRETE	PRIMARY	Differential Driver/Re- ceiver	TTL (0 - 3.5 volts)	64
(6)	DPR-4 (Discrete Interface)	TDS Uplink	Annunciator Response and Mode Switch LTS	DISCRETE	PRIMARY	Differential Driver/ Receiver	TTL (0 - 3.5 volts)	64
(7)	DPR-4 (Discrete Interface)	FDRS Television Subsystem	Pilot or Auto Selection of Background Scenes	DISCRETE	PRIMARY	NA	TTL (0 - 3.5 volts)	26
(8)	HW 716 (High Speed)	DPR-4 (High Speed)	Same as (1), (3), and (5) above	DIGITAL	PRIMARY for Proportion- al Data Backup for Dis- crete Data	Differential Driver/Re- ceiver	TTL (0 - 3.5 volts)	60K words/sec. 16-bit words
(9)	DPR-4 (High Speed)	HW 716 (High Speed)	Same as (2) and (6) Above	DIGITAL	PRIMARY for Proportion- al Data Backup for Discrete Data	Differential Driver/Re- ceiver	TTL (0 - 3.5 volts)	30K words/sec. 30-bit words

TABLE III. - CHARACTERISTICS SUMMARY OF THE INTERACTIVE GRAPHICS SYSTEM

HARDWARE CHARACTERISTICS	SOFTWARE CHARACTERISTICS
<ul style="list-style-type: none"> <li>● GRAPHICS-ORIENTED GENERAL PURPOSE CENTRAL PROCESSOR               <ul style="list-style-type: none"> <li>- 30 BIT WORD LENGTH</li> <li>- 1 <math>\mu</math> SEC CYCLE TIME</li> <li>- 32 K MEMORY ADDRESSING</li> <li>- 15, 30, AND 60 BIT ARITHMETIC AND DATA MANIPULATION CAPABILITY</li> <li>- 1 WORD FORMAT FOR 2-D VECTORS</li> <li>- 2 WORD FORMAT FOR 3-D VECTORS</li> <li>- INTERRUPT STRUCTURE</li> </ul> </li> <li>● DISK MEMORY SUBSYSTEMS               <ul style="list-style-type: none"> <li>- 40 AND 80 MILLION BIT STORAGE</li> <li>- STORES MONITOR AND OPERATING SYSTEM</li> </ul> </li> <li>● INTERACTIVE INPUTS               <ul style="list-style-type: none"> <li>- TABLET</li> <li>- LIGHT PEN</li> <li>- FUNCTIONS SWITCHES</li> <li>- KEYBOARD</li> <li>- VARIABLE DIALS</li> <li>- FOOT PEDALS</li> </ul> </li> <li>● 4 x 3 HYBRID COORDINATE TRANSFORMATION ARRAY               <ul style="list-style-type: none"> <li>- 3-D VECTOR CALCULATIONS in 4 <math>\mu</math> SEC - ACCURACY 0.1%</li> </ul> </li> <li>● GRAPHICS DISPLAY CAPABILITY               <ul style="list-style-type: none"> <li>- 3,570 CHARACTERS @ 40 FPS</li> <li>- 8,300½" VECTORS @ 40 FPS</li> <li>- UP TO 4 INDEPENDENT DISPLAYS</li> <li>- 4,160 1" VECTORS @ 40 FPS</li> <li>- 1,190 10" VECTORS @ 40 FPS</li> <li>- 32K x 32K POSITION GRID</li> <li>- INDEPENDENTLY ADJUSTABLE REFRESH AND FRAME RATES</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>● DISK RESIDENT OPERATING SYSTEM               <ul style="list-style-type: none"> <li>- FOREGROUND/BACKGROUND OPERATION</li> <li>- HIGH SPEED OVERLAY CAPABILITY</li> <li>- MACRO ASSEMBLY CAPABILITY</li> </ul> </li> <li>● SOFTWARE SUPPORT SYSTEM               <ul style="list-style-type: none"> <li>- FORTRAN IV COMPILER</li> <li>- GRAPHICS DISPLAY OPERATORS</li> <li>- ASSEMBLY LANGUAGE CAPABILITY</li> </ul> </li> <li>● GENERAL LIBRARY SUBROUTINES               <ul style="list-style-type: none"> <li>- FORTRAN OBJECT TIME SUPPORT</li> <li>- GENERAL MATH PACKAGE</li> <li>- INTERFACE I/O DRIVERS</li> </ul> </li> <li>● UTILITY AND SERVICE ROUTINES               <ul style="list-style-type: none"> <li>- SOURCE LANGUAGE TEXT EDITOR</li> <li>- SYSTEM SELF-TEST AND DIAGNOSTIC ROUTINES</li> <li>- DISK FILE MANAGEMENT PROGRAMS</li> </ul> </li> </ul>

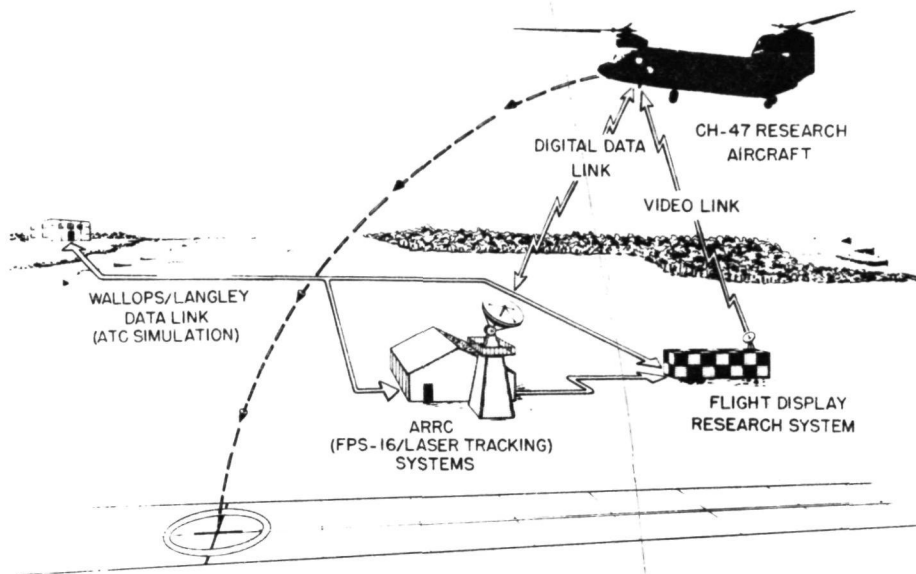


Figure 1.- Conceptual method of flight display testing.

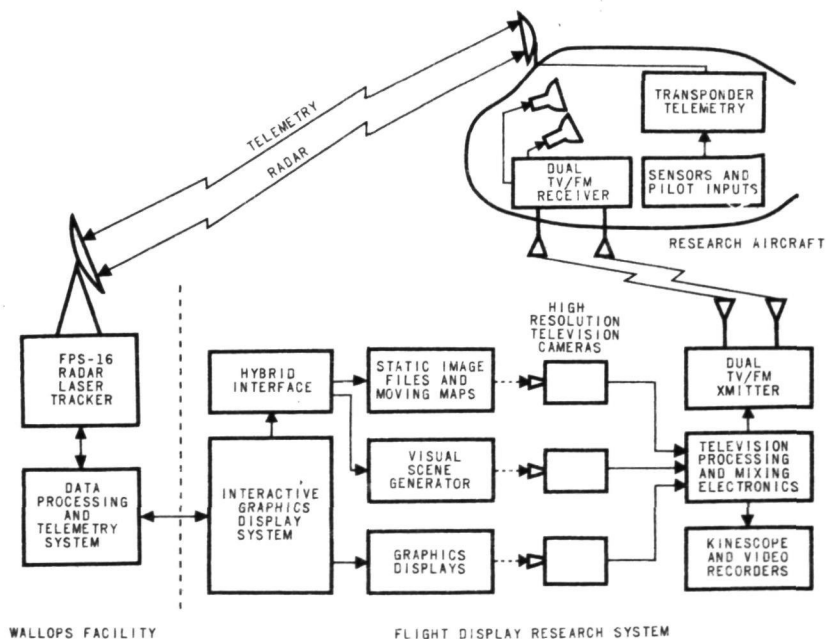


Figure 2.- Schematic diagram of flight display evaluation system.

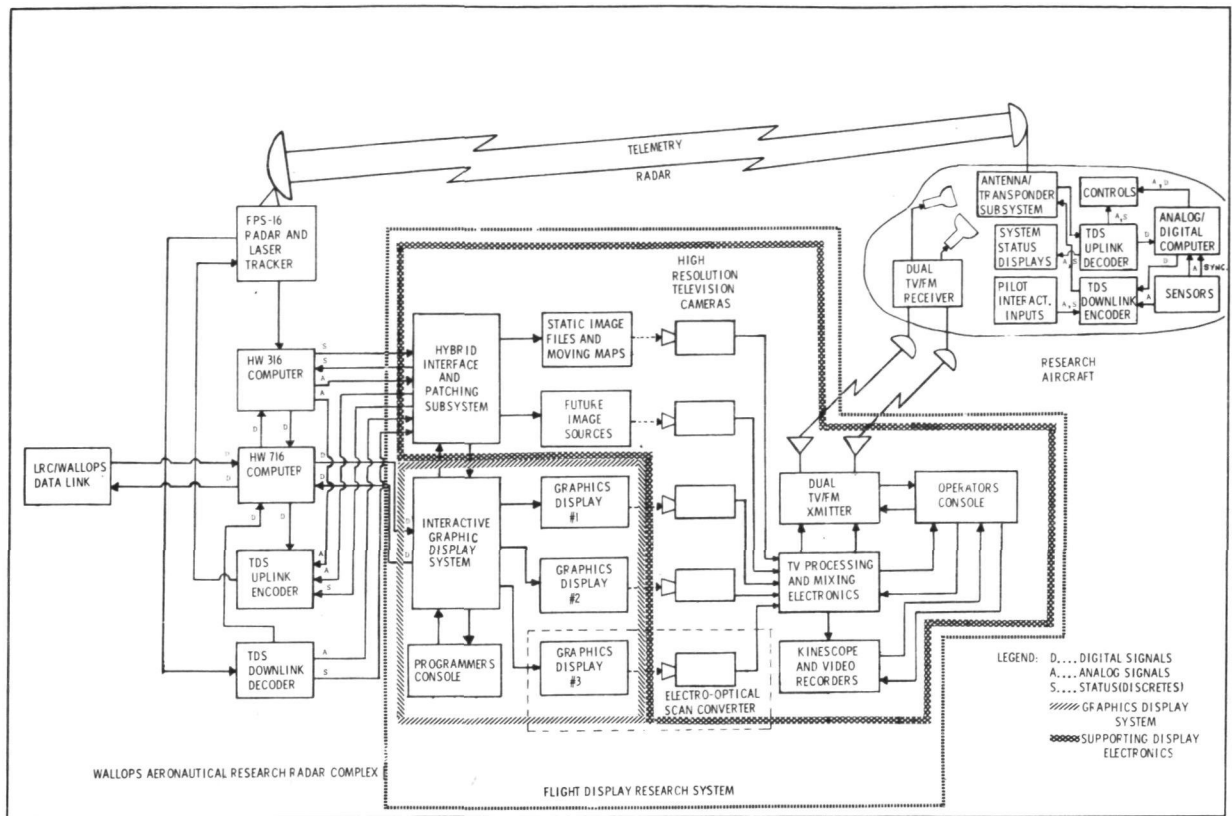


Figure 3.- Architecture and signal flow in the flight display evaluation system.



Figure 4.- Flight Display Research System.

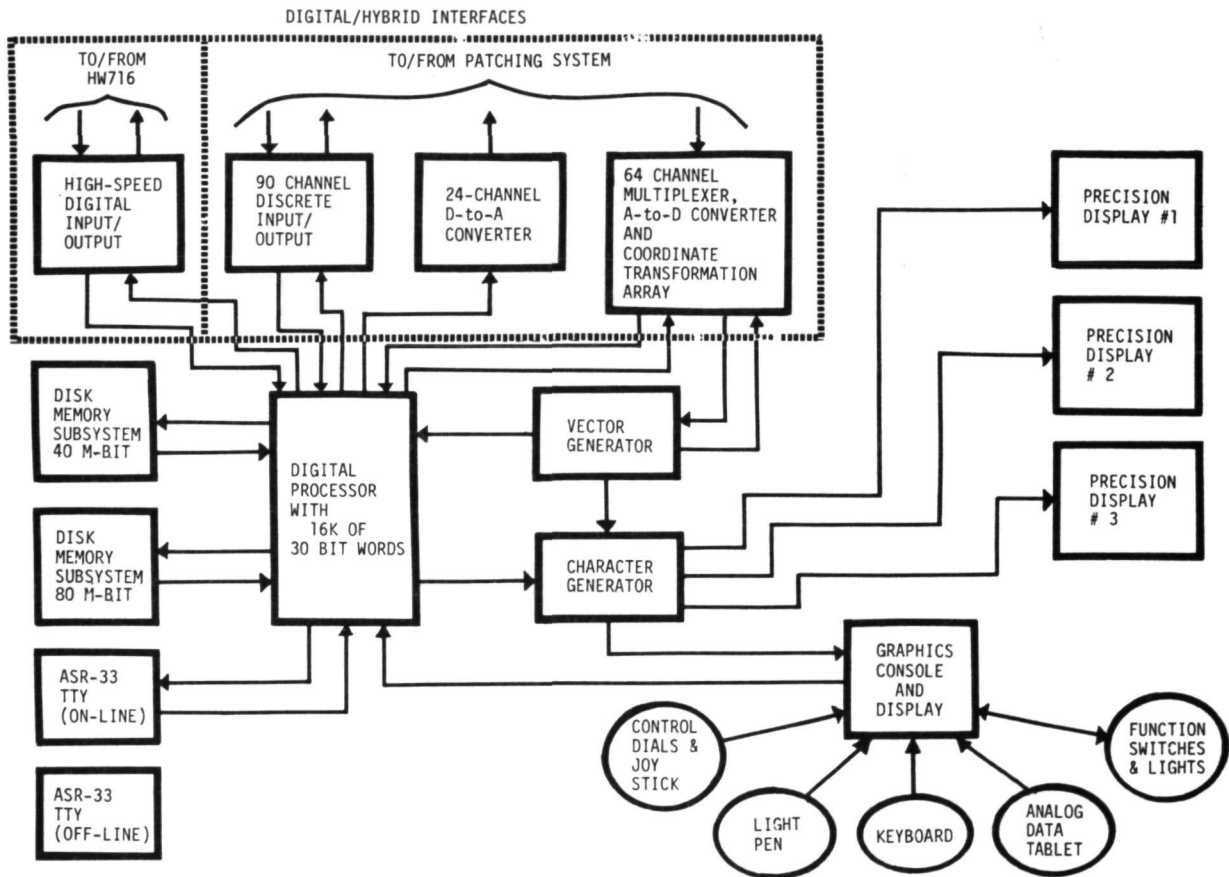
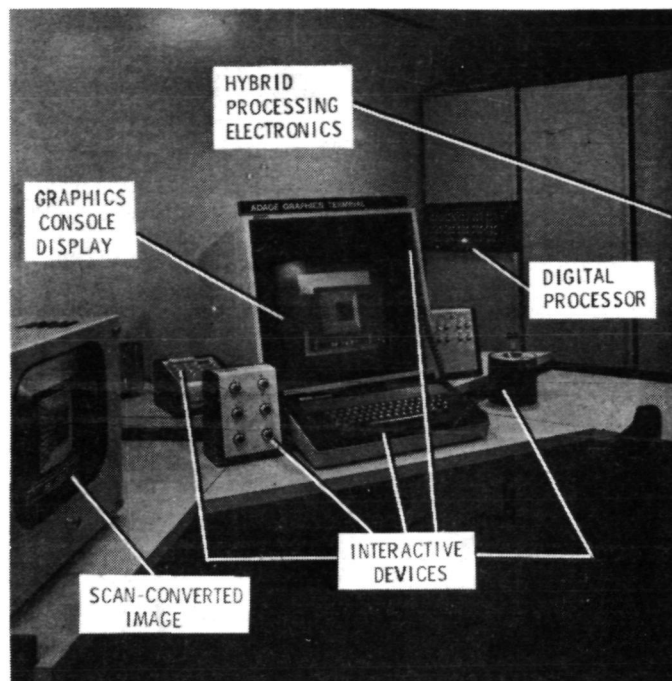
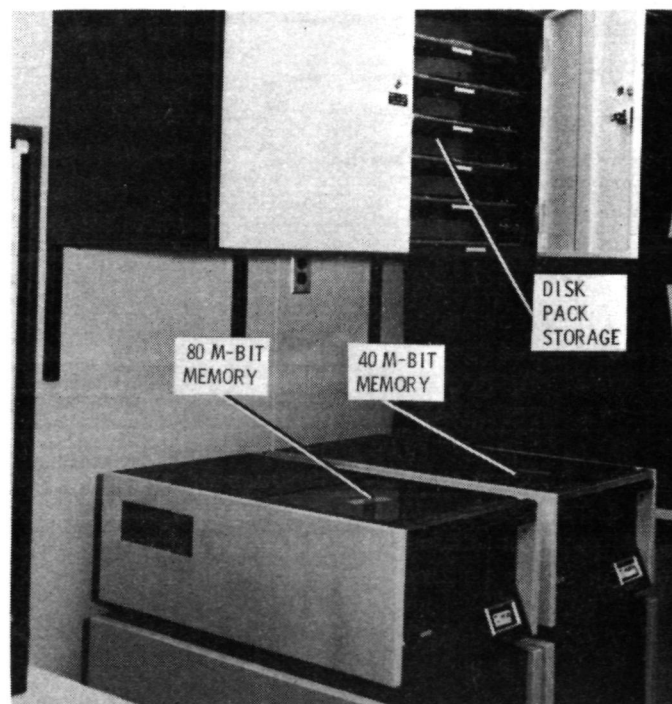


Figure 5.- Interactive graphics terminal schematic diagram.





(a) Interactive graphics terminal.



(b) Disk memory subsystem.

Figure 6.- Terminal and memory subsystem.

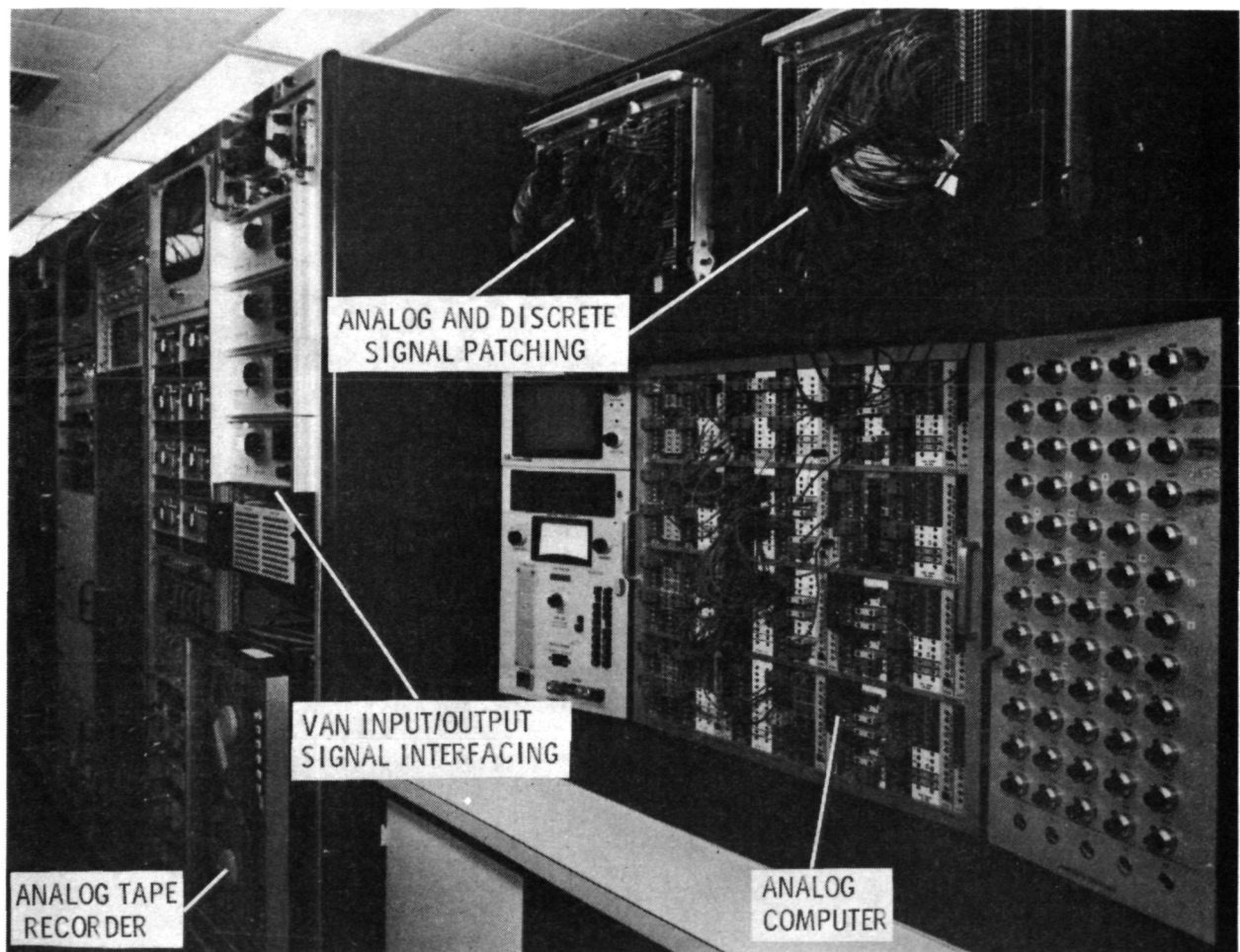
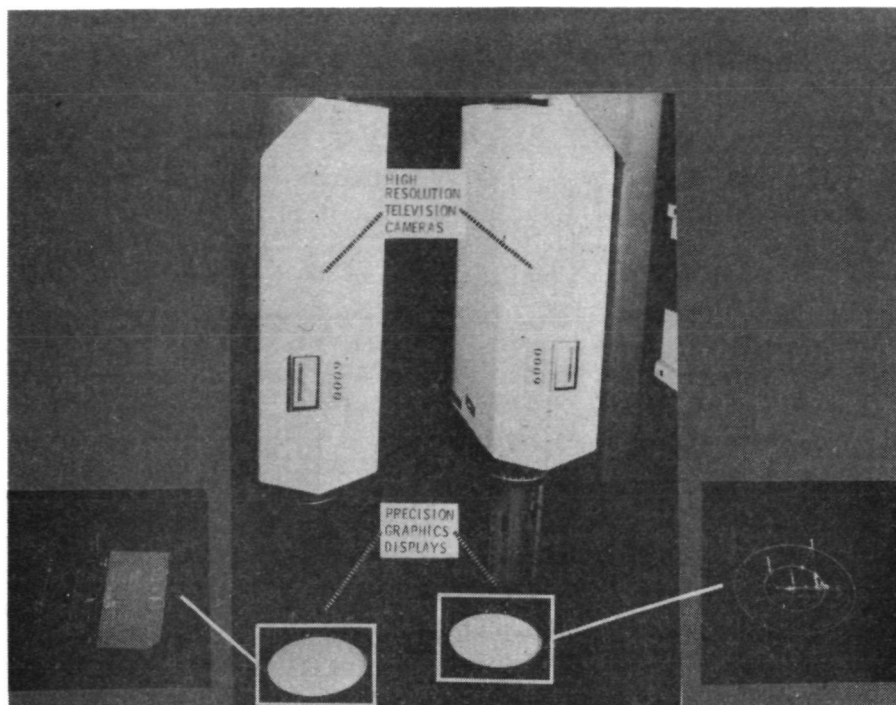
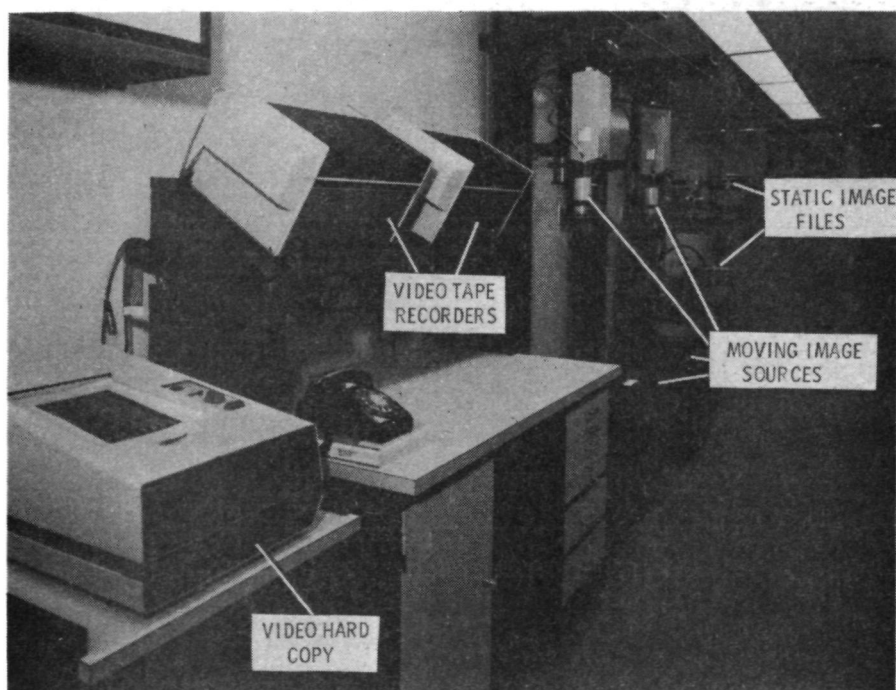


Figure 7.- Hybrid interface and patching subsystem components.

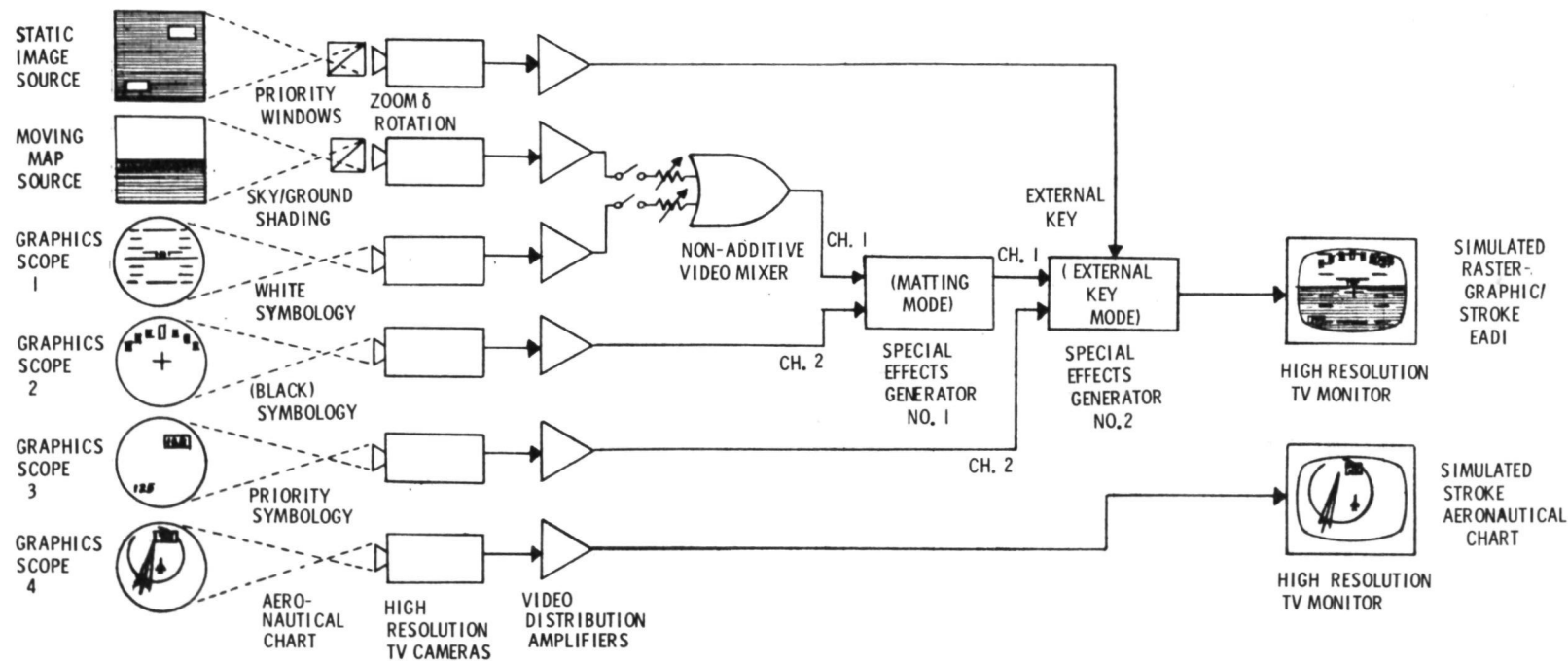


(a) Graphics-to-TV scan converters.



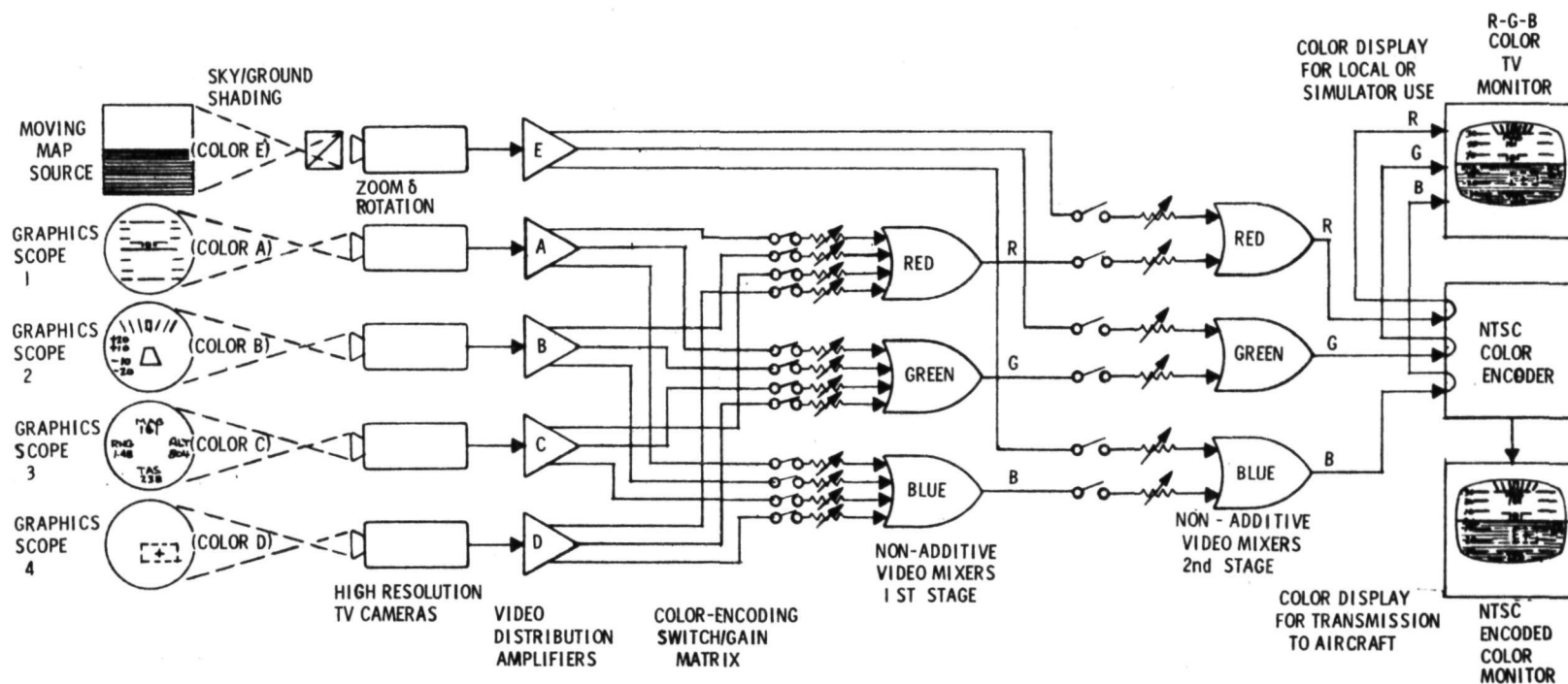
(b) Video imaging and recording components.

Figure 8.- Scan converters and video components.



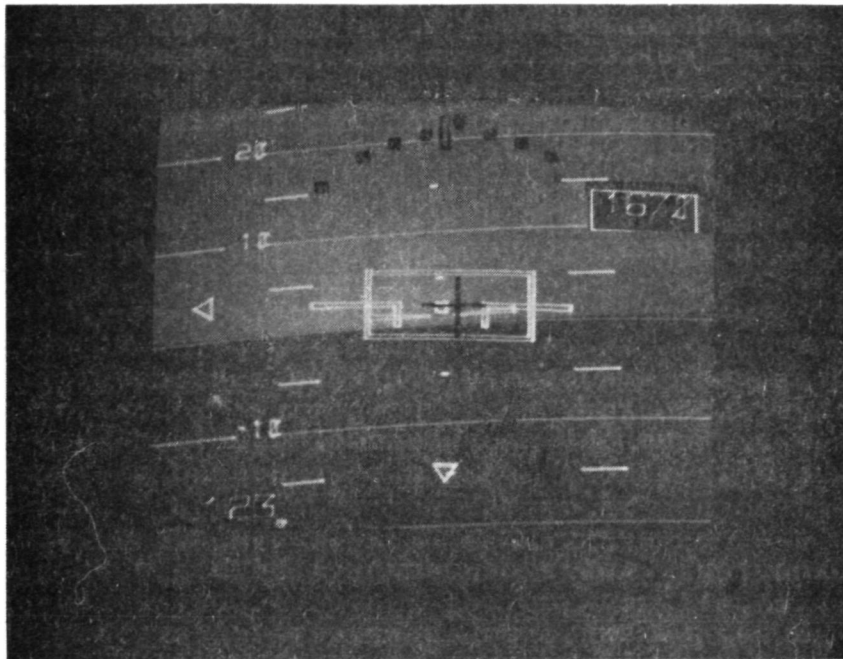
(a) Generation of a simulated rastergraphic/stroke-drawn EADI and a simulated stroke-drawn aeronautical chart, simultaneously.

Figure 9.- Video processing.

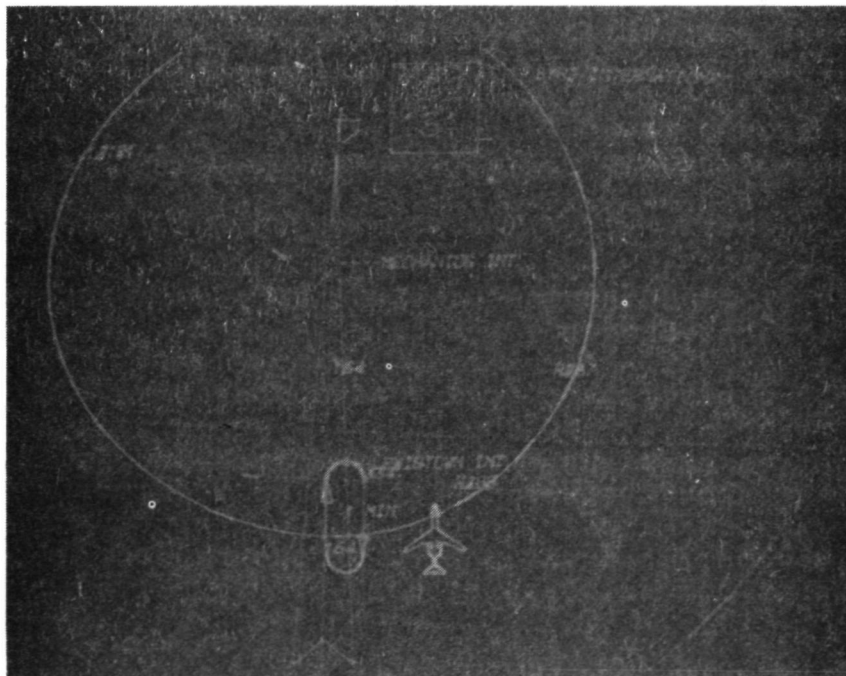


(b) Generation of a color-coded EADI display, with or without simulated rastergraphic sky/ground shading.

Figure 9.- Concluded.

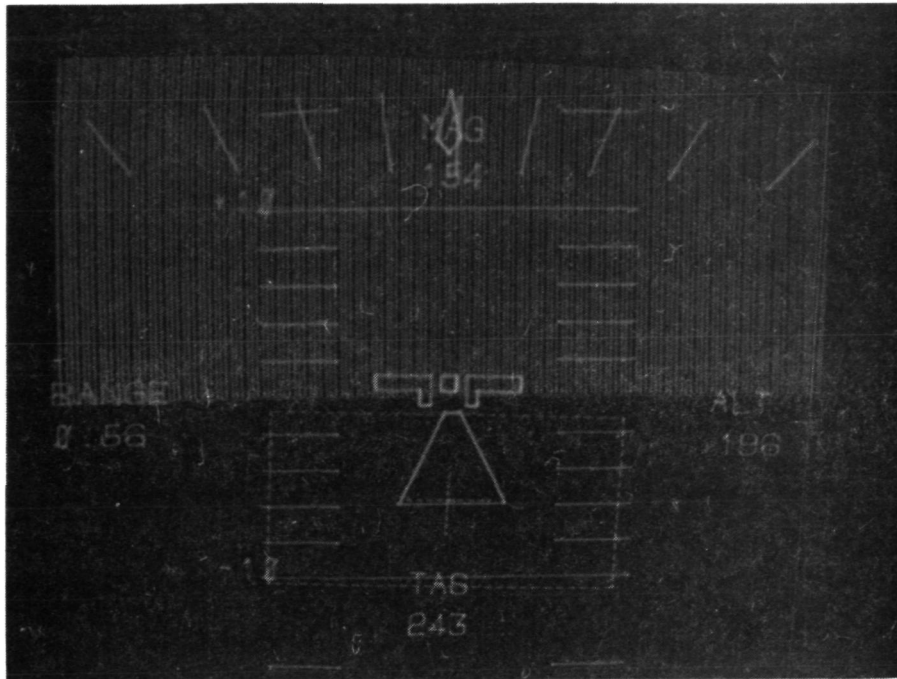


(a) Simulated rastergraphic/stroke-drawn EADI display, with sky/ground shading grey-scale encoded symbology, and priority-window symbology.

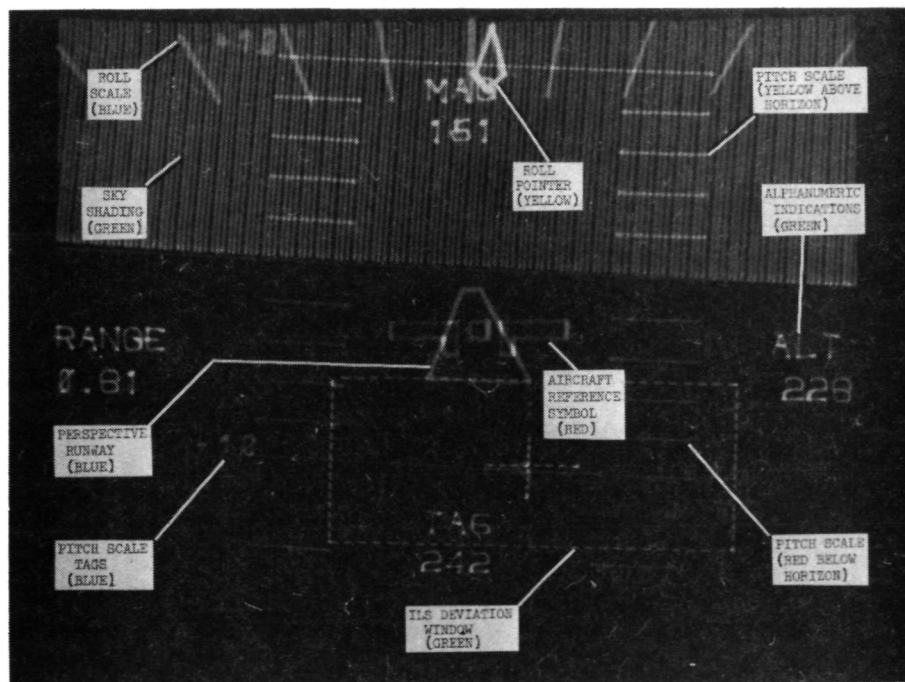


(b) Simulated stroke-drawn aeronautical chart display.

Figure 10.- Simulated displays.



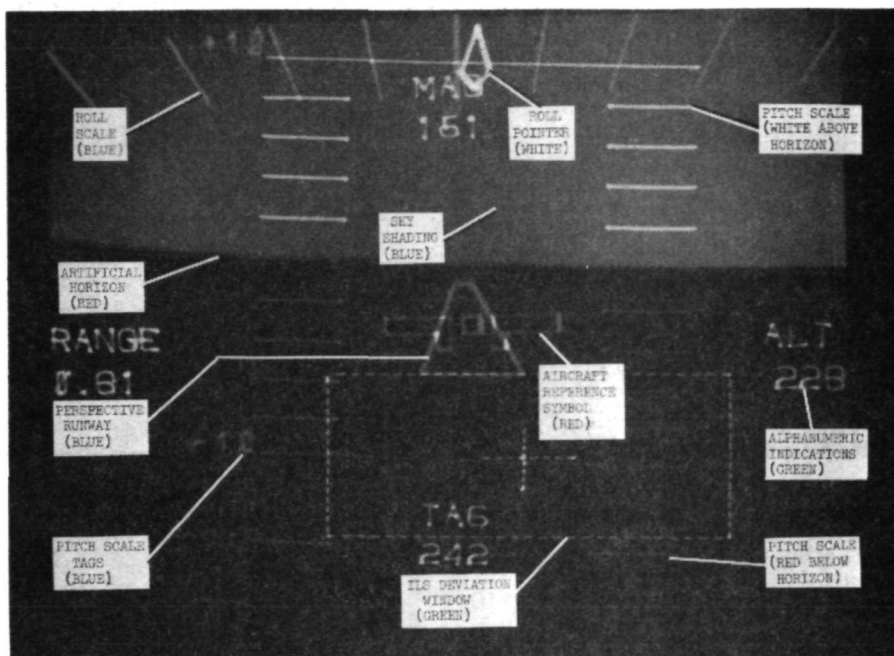
(a) Monochrome EADI display.



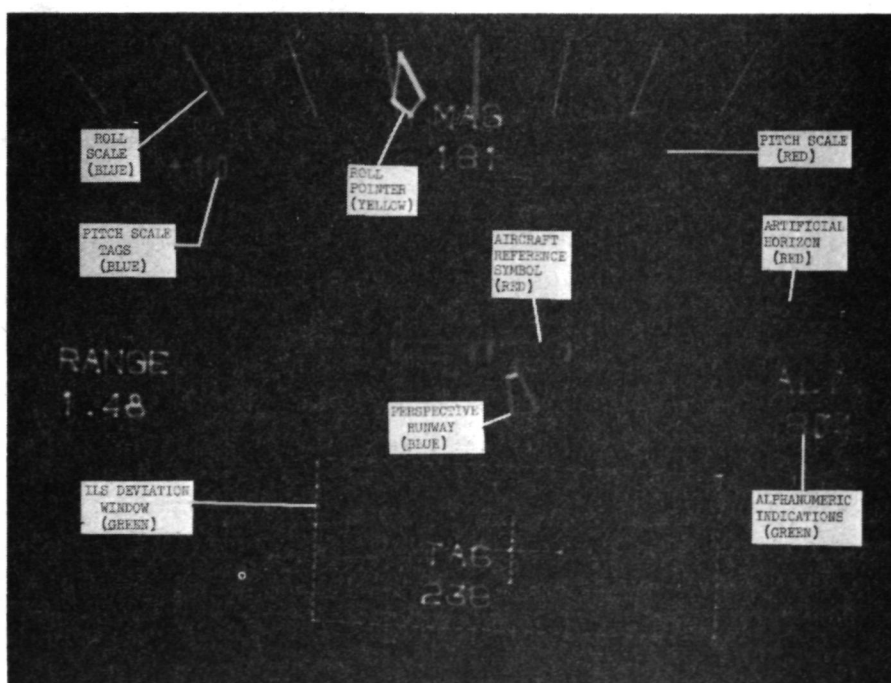
(b) Color-coded EADI display.

Figure 11.- EADI displays with stroke-drawn sky shading.





(a) With simulated rastergraphic sky shading.



(b) Without sky shading.

Figure 12.- Color-coded EADI display.



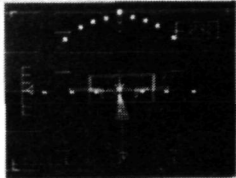
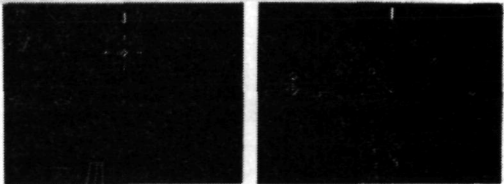
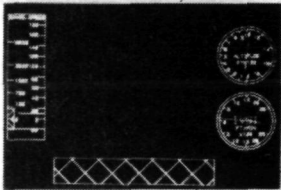
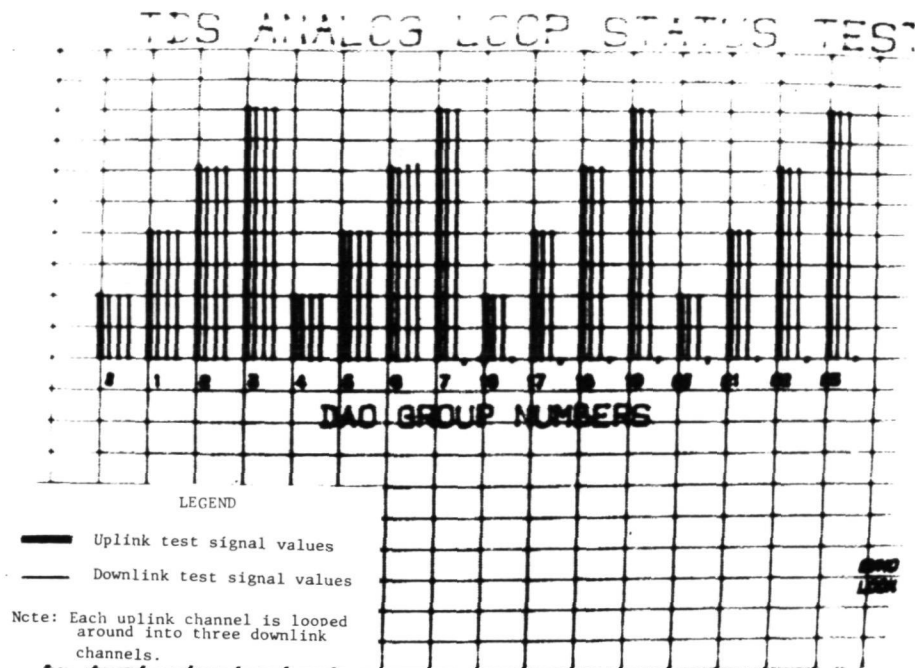
TYPE OF DISPLAY	PROJECT	IMAGE
ELECTRONIC ATTITUDE DIRECTOR INDICATOR (EADI)	TCV (737 SIMULATOR)	
ATTITUDE DIRECTOR INDICATOR AND INTIGRATED HORIZ./ VERTICAL SITUATION	VALT (CH-46/47 SIMULATOR/AC)	
VERTICAL SCALE, DIAL AND DRIFT- METER INDICATOR	VALT (SH-3A SIMULATOR/AC)	

Figure 13.- Displays generated by the FDRS for research simulation support.



(a) Diagnostic bargraph display of TDS data in the analog loop-around test.

### TDS/ANALOG SYSTEM LOOP ACCURACY TEST

DAO CHS OUTPUT	MUX GRP #1 IN	MUX GRP #2 IN	MUX GRP #3 IN
TEST VOLTAGE	-----LOOP VALUES, VOLTS-----		
DAO 0- 6.88	MUX16- 5.98	MUX32- 5.98	MUX48- 5.98
DAO 1- 6.88	MUX17- 6.88	MUX33- 6.88	MUX49- 5.98
DAO 2- 6.88	MUX18- 5.98	MUX34- 5.98	MUX50- 6.88
DAO 3- 6.88	MUX19- 6.88	MUX35- 6.88	MUX51- 6.88
DAO 4- 6.88	MUX20- 5.98	MUX36- 5.98	MUX52- 6.88
DAO 5- 6.88	MUX21- 6.88	MUX37- 6.88	MUX53- 6.88
DAO 6- 6.88	MUX22- 5.98	MUX38- 6.88	MUX54- 5.98
DAO 7- 6.88	MUX23- 5.98	MUX39- 5.97	MUX55- 5.98
DAO16- -6.88	MUX24- -5.98	MUX40- -6.88	MUX56- -5.98
DAO17- -6.88	MUX25- -5.98	MUX41- -5.98	MUX57- -5.94
DAO18- -6.88	MUX26- -5.98	MUX42- -5.98	MUX58- -5.98
DAO19- -6.88	MUX27- -6.88	MUX43- -5.98	MUX59- -5.94
DAO20- -6.88	MUX28- -6.88	MUX44- -6.88	MUX60- -6.88
DAO21- -6.88	MUX29- -5.98	MUX45- -5.98	MUX61- -5.98
DAO22- -6.88	MUX30- -6.88	MUX46- -5.98	MUX62- -5.94
DAO23- -6.88	MUX31- -5.98	MUX47- -5.98	MUX63- -5.98

Note: (1) Each uplink channel is looped around into three downlink channels.

(2) Left column shows values of uplink channels 0 to 15.

(3) Right three columns show values of downlink channels 0 to 15, 16 to 32, and 33 to 47, respectively.

(b) Diagnostic alphanumeric display of TDS data in the analog loop-around test.

Figure 14.- Diagnostic displays of TDS data.

# TDS/ANALOG SYSTEM LOOP ACCURACY TEST

DAC CHS OUTPUT	MUX GRP #1 IN	MUX GRP #2 IN	MUX GRP #3 IN
TEST VOLTAGE	-----ERROR, % OF F.S.-----		
DAO 1- 6.11	ERR16- 1.86	ERR32- 1.24	ERR48- 1.14
DAO 1- 6.11	ERR17- -1.84	ERR33- -1.24	ERR49- 1.24
DAO 2- 6.11	ERR18- 1.14	ERR34- -1.24	ERR50- 1.14
DAO 3- 6.11	ERR19- -1.86	ERR35- -1.86	ERR51- -1.18
DAO 4- 6.11	ERR20- -1.18	ERR36- 1.24	ERR52- -1.16
DAO 5- 6.11	ERR21- -1.64	ERR37- -1.64	ERR53- -1.54
DAO 6- 6.11	ERR22- 1.14	ERR38- -1.24	ERR54- 1.18
DAO 7- 6.11	ERR23- 1.24	ERR39- -1.64	ERR55- 1.24
DAO16- -6.11	ERR24- -1.40	ERR40- -1.40	ERR56- -1.63
DAO17- -6.11	ERR25- -1.63	ERR41- -1.40	ERR57- -1.63
DAO18- -6.11	ERR26- -1.42	ERR42- -1.41	ERR58- -1.52
DAO19- -6.11	ERR27- -1.47	ERR43- -1.42	ERR59- -1.63
DAO20- -6.11	ERR28- 1.20	ERR44- -1.37	ERR60- 1.17
DAO21- -6.11	ERR29- -1.40	ERR45- -1.42	ERR61- -1.40
DAO22- -6.11	ERR30- -1.40	ERR46- -1.33	ERR62- -1.52
DAO23- -6.11	ERR31- -1.13	ERR47- -1.13	ERR63- -1.23

Note: (1) Each uplink channel is looped around into three downlink channels.  
(2) Left column shows values of uplink channels 0 to 15.  
(3) Right three columns show values of downlink channels 0 to 15,  
16 to 32, 33 to 47, respectively.

(c) Diagnostic alphanumeric display of TDS data  
showing loop-error display mode.

Figure 14.- Concluded.

# THE APPLICATION OF INTERACTIVE GRAPHICS TO LARGE TIME-DEPENDENT HYDRODYNAMICS PROBLEMS\*

Fred Gama-Lobo and Lynn D. Maas  
Los Alamos Scientific Laboratory

## Introduction

This paper is a companion to a movie produced at LASL entitled "Interactive Graphics at Los Alamos Scientific Laboratory". The intent of the paper is to complement the movie. The movie presents the actual graphics terminal and the functions performed on it much better than any written description could possibly convey. However, the movie compresses and simplifies the processes being presented because the pace of a movie prevents detailed description. This paper attempts to put in perspective the complexity of the application code and the complexity of the interaction that is possible, which may not be apparent from the simple problem run as an example in the movie and the few simple examples of interaction presented in the movie. The paper, by itself, does not have as much of an impact, however, without the visual examples presented in the movie.

## Interactive Graphics System Description

The interactive refresh graphics system in use at LASL consists of a Sanders Associates, Inc. Model 960 display indicator, a Varian 620i minicomputer, and two Control Data Corporation (CDC) 7600 computers. The minicomputer controls two CRT displays, handles input devices, and communicates with the CDC 7600 computers on which the user programs run (see Figure 1).

A terminal consists of an alphanumeric keyboard, 16 system control keys, 16 lighted user function keys, a data tablet, a light pen, and a CRT. The minicomputer currently is supporting two terminals. One of these terminals has a single-color CRT with a display area of 14 inches square (1024 by 1024 resolution) while the other terminal has a four-color CRT (CPS, Inc. Penetration CRT) with a display area of 10 inches square (1024 by 1024 resolution).

---

\*This work was performed under USERDA Contract W-7405-Eng. 36.

The minicomputer communicates with the two CDC 7600 host computers via a multiplexed IO path at the rate of 3.2 megabits per second. The CDC 7600 host computers are running a mono-program batch operating system with modifications to allow interaction with the graphics system. Interaction is achieved by means of a high job priority and a disk rollin-rollout capability in the host computer.

This system was designed to provide interactive graphics for large, typically long-running programs that had previously been restricted to batch operation.

### Application Code

One code that was adapted to run on this system is a two-dimensional Lagrangian hydrodynamics code. The problems run by this code are set up as a two-dimensional matrix of points. These points define a mesh of quadrilateral zones that describe the geometry of the problem. The code carries the geometric information and all the physical attributes of the problem in doubly-dimensioned arrays. It carries two coordinates and two components of velocity for each mesh point and it carries a material identification, mass, density, temperature, pressure and other physical quantities for each quadrilateral zone. The indices to these arrays are called K and L by convention. For a constant K, points sequentially indexed by L are connected by line segments and a sequence of such connected points is called a K line. For a constant L, points sequentially indexed by K are connected by line segments and a sequence of such connected points is called an L line. The intersections of K and L lines form the quadrilateral zones. By convention, the zone quantity arrays indexed by K and L refer to the quadrilateral connecting the points  $(K,L)$ ,  $(K-1,L)$ ,  $(K-1,L-1)$ , and  $(K,L-1)$ .

The code computes the motion of the problem in time by breaking the problem up into discrete time steps. The following is a simplified description of the computational cycle for each time step. First, the code establishes a time step for the cycle based on various stability criteria. The code then uses numerical difference equations based on analytic equations describing the physical phenomena occurring in the problem to determine the motion of each mesh point for the cycle. This is essentially done by differencing the pressures of the zones surrounding a point to determine the change in velocity for the point during that cycle and then multiplying the velocity by the time step to determine the change in position of the point. The mass of each zone does not change with time, so the new volume of each zone is used

to compute a new density. Tabular equations of state are then used to get a new pressure for each zone. The code then goes through the same cycle for the next time step.

### Rezoning

An annoying problem with Lagrangian hydrodynamics codes is mesh tangling. As was mentioned before, the motion of each point is calculated independently. One would hope that if the equations used are correct, all the zones would stay regular. However, the complex motions of the mesh can cause some of the zones to get very distorted. These distortions can sometimes cause problems, such as zones getting artificially high temperatures and pressures because the zones are unrealistically compressed or the distorted zones causing the problem to run at a lower than normal time step, thereby requiring more computer time to complete the problem. There are also unphysical distortions that occur when a mesh point somehow gets a spurious velocity that causes it to cross over into a zone outside its surrounding zones. The reason such problems occur is not always known, but it is probably because of trying to describe a continuous medium with a finite number of zones.

The remedy to these problems is rezoning. This is done by moving points to reduce the distortions. The new mesh configuration is then mapped onto the old mesh to redistribute the physical quantities, such as mass and energy, based on the intersections of new zones with old zones.

## Interactive Techniques Used in the Code

### Rezoning

The most significant application of interactive graphics in the code is in rezoning. The rezoning process deals almost entirely with graphical data and is done very naturally at the graphics terminal. The easiest way to discuss the value of interactive graphics in this application is to compare the rezoning process prior to the availability of interactive graphics with the current techniques.



The old rezoning process was as follows. First, a drawing of the mesh was obtained by running a restart dump tape through a post-processing code which generated a plot on a Calcomp plotter. This mesh plot was then examined to determine the area where tangling occurred. At this point, it was often necessary to get another mesh plot of an enlarged window around the area of tangling, because such tangling usually occurs where zones are being crushed and points are so close together that they are not resolved in a drawing of the full mesh. In order to define such a window, it was necessary to look up coordinates of points in a listing, sketch the desired window on the drawing, and measure relative distances to the known points to determine the coordinates of the window limits.

At the graphics terminal, a series of windows can be examined in succession and the area of mesh tangling can be quickly isolated. No coordinates need be known because the user can define the window he wants from the full mesh display on the screen by using the data tablet to point to the positions of the window limits. If the resulting window is still too big to resolve the tangled lines, he can continue defining smaller windows until he does resolve the tangled lines.

Rezoning is accomplished by moving mesh points to reduce distortions. There are several input commands to the rezone code to do this. The most natural method is to move a point by giving it a new set of coordinates. With card input, the coordinates have to be punched on a card. Determining the coordinates needed to move a point can be complicated if one has to extract this information from a mesh drawing. It is easy enough to mark the new position of the point on the drawing, but it is then necessary to measure the components of the distance of that point to an existing point in the problem, look up the coordinates of that point in a listing, scale the measured components from the drawing to the problem coordinate system, and then add them to the coordinates of the existing point. Since this is so cumbersome, people tended to use other available commands that do such things as equally space lines, straighten lines, and map the proportional spacing along one line onto another. The command to equally space a line, for instance, needs only the indices of the two end points of the line specified.

At the graphics terminal, it is very easy to move a point. The user points to a mesh point on the display and then points to the new location, using the data tablet. The display then changes and the mesh point moves instantly to its new location. Commands such as equal space are also simplified. Determining indices of points from a mesh drawing can be a nuisance, too, when dealing with a large problem having many

points. At the graphics terminal, the user points to the end points of the line to be equally spaced, using the tablet. Again, the display changes instantly to show the results of the equal space command.

This brings up another advantage of interactive graphics. The entire rezoning process takes place sequentially with each step producing a new picture. In the batch mode, the entire rezone would have to be predetermined prior to submitting a rezone run. This was usually done by sketching on the mesh drawing. If the rezone was complicated, then the drawing would become very confusing, thereby necessitating several batch submittals of rezone runs, each requiring a post-processing run to generate mesh drawings. It was not unusual for a rezone to consume several days, and therefore people avoided doing them.

There is a class of problems that will not run without a large number of rezones. Some problems run recently have required some rezoning about every 5 minutes of CP time, with the problem running a total of 5 hours of CP time. Such problems were usually not run prior to the interactive graphics capability, and if they were, they would have taken several months. Now they can be done in four or five terminal sessions of approximately two hours each.

### Monitoring

The other key interactive feature of the code is the capability of monitoring the execution of the code and interrupting at any time to rezone. The execution is initiated by typing RUN at the keyboard. While the problem is being monitored, successive pictures are displayed on the screen at time intervals specified by the user. In this mode, the code remains resident in the host computer and just continues running. The user can stop the run at any time by hitting a fuction key. The user can then examine the mesh and various problem values printed on the screen and he has the option of rezoning or continuing to run the problem. This capability is essential to give the user total control of the calculation at the terminal.

In the initial implementation of interactive graphics in this code, the rezone package was a separate code. It ran interactively, while the main computational code still ran in the batch mode. There was an immediate payoff in the



simplification of rezoning, but for problems requiring 50 rezones, that meant submitting 50 batch computational runs and scheduling 50 terminal sessions to do the rezoning.

This was not the only drawback. The way the computational code communicated with the rezone code was by restart dumps on tape taken at time intervals specified by the user. Quite often these dumps were not available at the best times for rezoning. It was not unusual to get a batch run back in which the mesh looked good for one time dump and was so badly distorted on the next dump that the rezoning required became very complicated. The monitoring capability lets the user pick the optimum rezoning times. He can do this visually by looking at the mesh or by examining various quantities printed on the screen which may indicate that there are problems arising in the calculation.

It is this monitoring function which necessitates the link to a machine with the great computational speed and capacity of a CDC 7600. For typical problems, the meshes range from 5000 to 10000 zones and describe very complicated geometries. A large memory is required to hold the many physical quantities stored for the mesh. Also, the numerical difference equations used by the code to describe the physical processes being studied with the aid of this code are very complex and require the computational speed of the fastest computers currently available to run problems in a reasonable amount of time. Typical problems can consume from 5 to 10 hours of CP time. The rezoning, by itself, could be done by a much slower computer, but the capability of total interactive control of a calculation that the monitoring function, along with the rezoning, provides, absolutely requires that the graphics terminal be linked to the fast computer.

The batch operating system on the host computer turns out to be an advantage for this application. This class of code generally does not compete well in a time-sharing environment. During a terminal session, a user will typically consume an hour of CP time, consisting of five minute chunks of computational time separated by rezones. For the user to make efficient use of his time at the terminal, those 5 minute computational chunks requiring the full capacity of the host should be executed without interruption. On this system, that is possible. The number of possible terminals is severely limited on such a system, of course, but in a computing environment consisting of many large, long-running codes, the number of terminals must be limited.

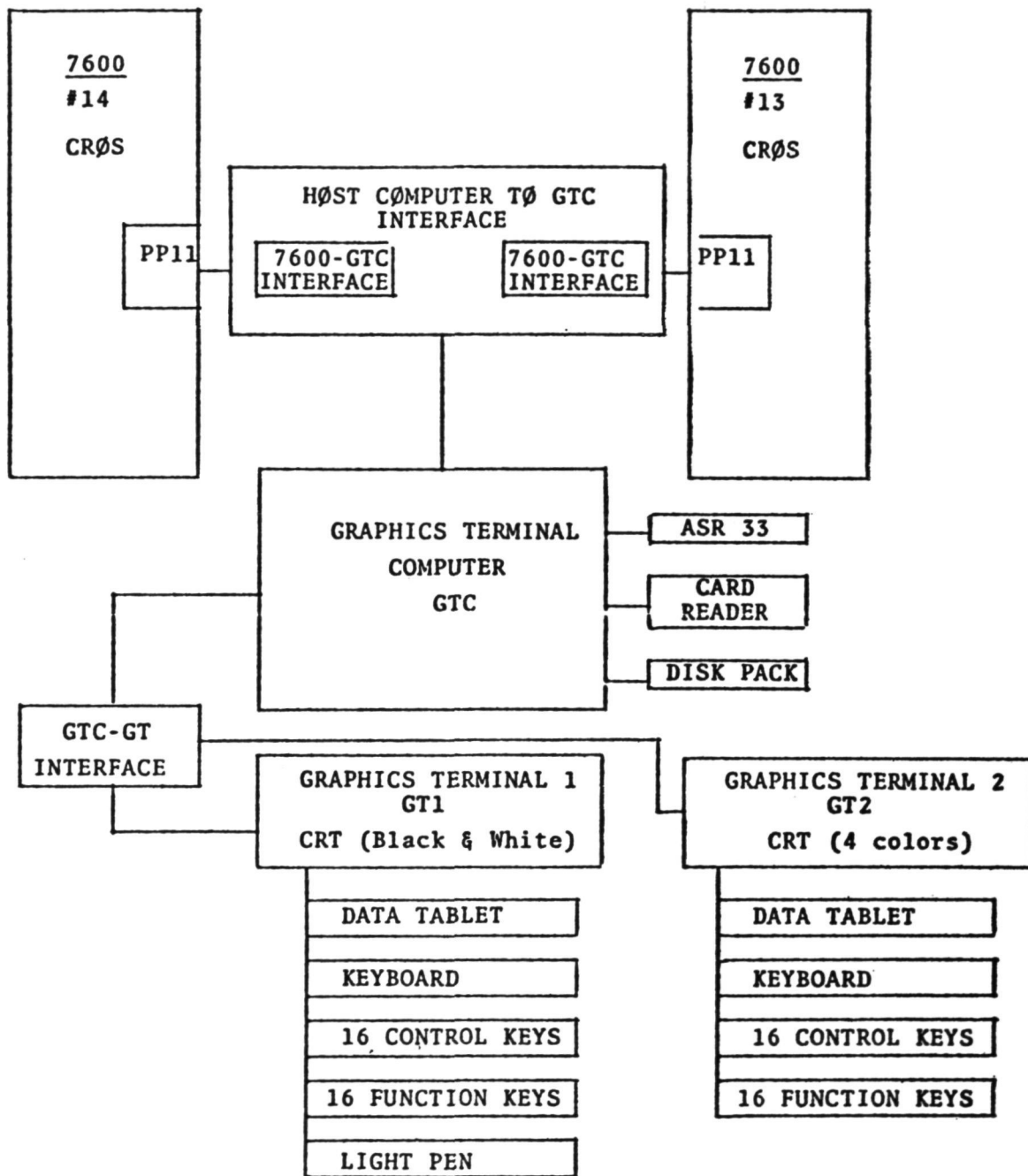


Figure 1

Hardware Configuration of LASL Interactive Graphics System

Page Intentionally Left Blank

# COMPUTER GRAPHICS FOR MANAGEMENT

## An Abstract of Capabilities and Applications of the EIS System

Barry J. Solem

Boeing Computer Services, Inc.

### PREFACE

The object of the Executive Information Services (EIS) System is to change the way businesses are managed. That is being achieved by several hundred business professionals who are using the EIS language to solve their own problems. On the basis of that experience, it is clear that the course of business will change when decision makers regain a measure of control over an appropriate computing resource!

The system provides a wide variety of services including computational, data base, and display capabilities. Computer graphics is only one of those resources. The attractiveness of EIS graphics is in their ease of use and in their relationship to the rest of the system. For that reason, the only practical way to understand the application of EIS computer graphics is to understand the system as a whole.

### BACKGROUND

The problems associated with exploring, making and implementing management decisions are among the most challenging assignments in business and government. They are frequently time-constrained, relatively unstructured, and almost always require some degree of participation or direction from management. Although they usually have a big impact on the organization's success, managerial prerogatives are often constrained by the absence of good information and an insufficient number of alternatives.

Historically, computing services have made less of an impact on this area than in the high volume, and repetitious functions of accounting, payroll, scheduling, etc. The reasons are plentiful. One of the principle factors has been tradition. The first computer programs were extensions of work performed on tab and accounting machines and by large organized groups of people. Over

the years, the rapid development of equipment and the opportunity to do a better job in this same area has captured the largest part of the computing work force. With all that experience, systems people have become quite skilled and a bit comfortable working the same set of problems.

Another cause for the slowness of response is communication. Early computing hardware and software were difficult and sometimes frustrating to work with. The groups that gathered to solve the problem were more oriented to computing than the activity of management. The rapid growth of technology has served to further isolate the two groups. Ironically, a third cause of delaying computer service to decision makers is certainly a result of previous attempts in that direction. A history of past failure has produced a community of discouraged computer and business leaders who have convinced themselves that it can not be done.

During the last several years, tremendous changes have been taking place within the computing world. These changes are at the roots of a new relationship which is emerging between the data processing and business sectors. The driving force behind these new relationships is, of course, the accelerated development of hardware technology and the resulting rapid decline in costs. Time-sharing, mini computers, low-cost terminal and graphic devices are among the products of this technology. One of the important by-products of time-sharing and mini's is the interest in computers which is springing up from outside the inner circle. Whether welcomed by the computer fraternity or not, that interest will result in the active participation of engineers and businessmen, in solving their own problems. The combination of events will, without a doubt, create an environment within which some break-throughs in computing service will occur. Among the most important of these will be broad and productive use of the computer by business decision makers.

## SYSTEM OVERVIEW

### Objectives

The Executive Information Services (EIS) capability was initiated by The Boeing Company in 1969. From the beginning, the emphasis was: (1) to create a system which would extend computer resources to persons involved in the business decision process, and (2) to provide correct and complete information to these persons so they could make good decisions based on fact. After an extensive survey, the following conclusions began to emerge:

- o The system had to be geared to the interactive nature of target problems (which were usually solved by groups of people and desk calculators).
- o The system had to be usable by persons within the business functions. The requirement for specialized knowledge and quick response usually precludes the involvement of a third party.

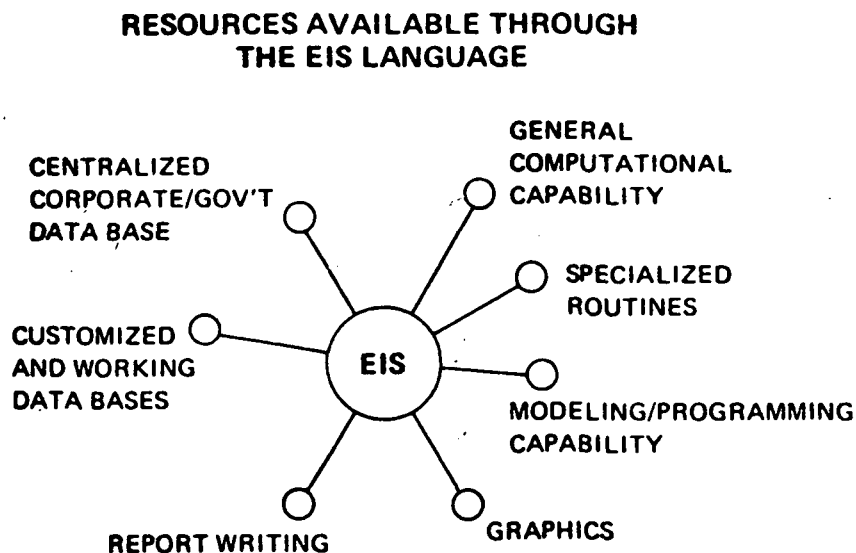
- o The system needed to be flexible because the problems were varied and unpredictable.
- o The large data collection needed to be organized into specially designed data bases with an emphasis on ease of use and the unique characteristics of the data.

Shortly after the EIS System became available, we discovered the following important facts which shaped all subsequent development:

- o Unless a system is useful on a daily basis to solve a majority of a businessman's problems, his computing skill will quickly fade and his interest in the system will fade with it.
- o A single system should be broad enough to support users across functional and divisional boundaries. Mobility of staff demands it.
- o Development of an effective management support system is never finished. Technology, new opportunities, change in problems and growth in the capacity of users require a continuing response.

#### General Characteristics

The EIS System was developed to provide a broad spectrum of services for direct use by persons engaged in the various business professions. It is available to users anywhere in the 48 states through Mainstream-CTS (the Boeing Computer Services' commercial time-sharing service) on an IBM 370/168.



**FIGURE 1**

EIS resources include a full range of mathematical capabilities, graphics and report writing services, data base capabilities designed for ease of use and the types of information most frequently used in managing resources, and a number of other special features (see Figure 1). All of these are integrated into the system in a complimentary way and are available through the EIS language. Because the system is designed as a language and not a narrowly-defined program the user can combine the instructions to do whatever his needs require. Although the commands referenced in this document are written as full words, all EIS commands can be abbreviated by their first two characters.

To date, hundreds of people have been trained in the EIS language. These include persons within and outside The Boeing Company and persons with every type of educational and functional experience. Some of the capabilities they use were developed by the EIS Staff for general use, but most routines were developed with the EIS language directly by business, engineering, and management personnel to solve their own problems. The system is written in Fortran and BAL, but the largest volume of activity is serviced by programs written in assembly language so the cost of use is minimized.

### Data Types

One of the foundations of EIS is a variety of data types which respond to the unique characteristics of business operations. The three most commonly used are timespread, unit and cum data. Timespread data is defined as a data set for which a beginning date is established and which contains one or a series of values associated with a sequence of months, quarters, half-years, or years. The beginning date is usually established by month and year, although it could also be defined by day or week. Timespread data is often used in estimating and planning, economic or business trends and performance measurements. The Consumer Price Index for example is a set of monthly indices which show the changing cost of goods and services over time.

Unit data are one or a set of values associated with a sequence of events--usually production articles. They are similar to timespread data except the reference point is a unit position. These data sets are most often used to store the value or schedule date for high-value manufactured goods. In the case of both time and unit data, the reference date or unit makes it possible to perform calculations without any concern for the length or beginning point of the data sets used. Cum data are simple scalar or matrix values without time or unit associations. For example, one cum value could represent the sum of several months or units of activity.

Within the system, all information except that contained in special title files or data bases are identified by names which the user assigns.

## CREATING DATA

The principle methods of creating data are: (1) inputting information from a terminal or data files, (2) general and specialized computation capability, (3) data retrieval, and (4) computational services within customized user data bases.

### Input

Information can be entered into the computer from either the terminal or a data file through the INPUT command. In each case, the 'IN' instructs the system to accept the associated alpha-numeric information as an EIS data set. The first example is a cum data set with 11 values and is being INPUT into EIS under the name, X1. The second and third are examples of timespread data which were arbitrarily named, HRS and RATE. The TI in the second and third examples defines the data as time ordered. The numbers in parentheses establish the month and year for the first value.

```
IN X1 22 20 33 31 25 24 21 22 25 29 33:
```

```
IN HRS TI(1,75) 33 44 55 67 79 87 90 82 63 53 52 58
                64 68 70 66 55 41 33 29 27 26 25 24:
```

```
IN RATE TI(8,74) 5*5.26 12*5.91 12*6.47 12*7.10:
```

The asterisks (\*) in the last case imply repetition. In the example, 5\*5.26, \$5.26 rate applies to the five-month period from August through December 1974.

### Computation

The most common method of creating new information is through computation. The COMPUTE command supports the standard operations of addition, subtraction, multiplication, division, and exponentiation, plus a number of special functions. In the first example, a new data set 'COST' is created by multiplying the values of HRS times RATE (both referenced above).

```
COMPUTE COST = HRS * RATE
```

```
COMPUTE CTD = CUM(COST)
```

```
COMPUTE X2 = X1/TCUM(X1)*100
```

The two data sets being multiplied are time oriented, but differ in both starting data and number of values. The resulting data set (COST) will contain 24 monthly values and begin in January 1975.

In the second example, the functional operator 'CUM' is used to develop the running cum or cum-to-date values for data set 'COST'. The value in the second position of 'CTD' will equal the sum of the first two values of 'COST', the third month value will equal the sum of months one through three of 'COST', etc.



In the third example, each of the nine values in X1 will be divided by the total cum of all X1 values and multiplied by a constant 100. The resulting X2 will contain the percent that each element of X1 is of the whole.

The special functions include many of those available in other programming languages plus a number of others which allow users to perform various cumming operations, to move data sets forward and backward in time and to create moving averages.

### Specialized Computation

EIS also provides a number of subroutines to service some of the unique requirements of business and management. In every case, these special routines are activated by the first two characters of the command and followed by the appropriate arguments. With a SPREAD command, for example, a new data set may be created, which has the same expenditure pattern as another data set, but which may differ significantly in magnitude and duration. In the first example, the timespread data set 'NEW' will be created with 50,000 hours (dollars, etc.) distributed over a 36-month period beginning in September 1975.

SPREAD NEW TI(9,75) OLD 50000 36:

LEARN X3 80 1 10000 1 200:

The data sets, 'NEW' and 'OLD', will be the same general shape but may differ in every other respect. In the second example, a unit cost data set, X3, will be created using an exponential formula which approximates the experience observed in some manufacturing processes. X3 will be said to have an 80% slope. The first value of X3 will be 10,000, the second will be 80% of the first, the fourth will be 80% of the second, etc. The last two arguments will result in computation of values for Units 1 through 200. The unit positions and values for the first 15 are shown below. Numerous other special routines are available.

#### X3

1	10000.	2	8000.	3	7021.	4	6400.	5	5956.
6	5617.	7	5345.	8	5120.	9	4929.	10	4765.
11	4621.	12	4493.	13	4379.	14	4276.	15	4182.

## Work Functions

A third type of computation service is known as work functions. Several dozen are available. These provide services ranging from the creation of random numbers to return on investment and depreciation routines. An especially important feature of work functions is that users may also write (or have written) work functions in Fortran or BAL to do any task which their work requires. These routines may be independent or they may operate in combination with any other part of the EIS software. These user-written routines then become an integral part of the user's personal EIS version. The availability of work functions gives the user considerable control over the efficiency and versatility of the system to handle unique problems.

## Data Bases

Another method of creating information is through interface with an EIS data base. This can be done through the retrieval of information from an existing data base or by inserting information into a hierarchical data base. In the latter case, additional information may be generated within the data base itself. These will be discussed in a later section.

## DATA STORAGE

All of the information created in a terminal session through data input, computation, data base retrieval, is preserved in core and on disk for the duration of the EIS computer session. These are available through their user assigned names for further computation and display. At the end of each terminal session, these data sets are automatically purged to avoid confusion with work done at a later time.

Information can be saved permanently or for use in a later session by identifying the data sets to be preserved and a file in which they will be stored. The command:

SAVE /XYZ/ HRS RATE COST:

would result in three data sets, 'HRS', 'RATE', and 'COST', being saved in a file named, XYZ. If no data set names are identified, all of the information currently defined will be saved in the specified file.

Information can be reclaimed from a SAVE file for use through a READ instruction. The READ command:

READ /XYZ/ HRS COST:

will bring data sets, 'HRS' and 'COST' into active core from file, XYZ. If no data names are listed, all of the information within the identified file will be brought into active core.

A third form of data storage are structured EIS data bases.

## EIS DATA BASES

### Retrieval

A data base is a collection of information stored in a systematic way for use in one or more applications. EIS data bases are multi-dimensioned structures, especially designed to support business and management applications and within which time, unit, or cum data sets are stored. Data bases for each of these data types are organized in a similar way. For that reason, the capabilities of EIS data bases will be illustrated with timespread data.

EIS data bases vary from a few hundred to several million combinations and from simple two-dimension to complex multi-dimensioned structures. In every case, however, data is defined through the selection of element(s) from each dimension. To illustrate the process, it is convenient to consider the retrieval of data from a cubic structure whose dimensions we will call HEIGHT, WIDTH, and DEPTH. The user could then recall information as follows:

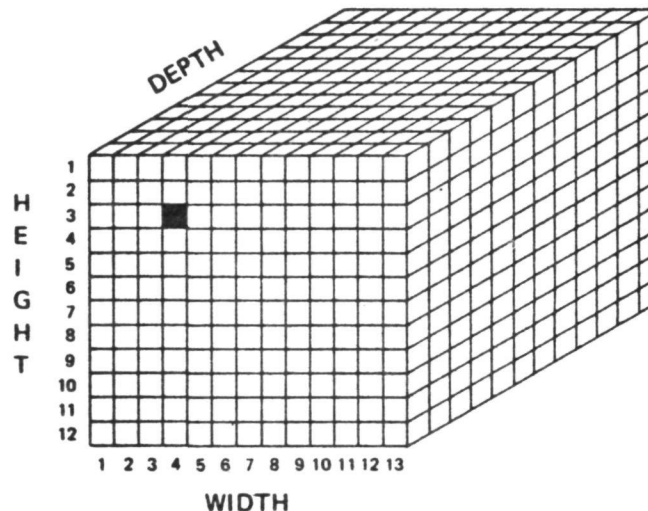


FIGURE 2

HEIGHT 3

WIDTH 4

DEPTH 1

GET A1:

The first three instructions define the characteristics of the required information. The fourth command causes information to be retrieved and stored under the name, A1. Time or unit data would be retrieved as follows:

GET A2 TIME:

GET A3 UNIT

Normally, it is more convenient and meaningful to assign names to the dimensions and elements which are more descriptive of the data base content. Consider, for example, a multi-product, profit center for which we wish to measure profit, cost, and revenue over time. The relevant factors might include the following products and financial factors:

<u>PRODUCTS</u>		<u>FINANCIAL FACTORS</u>	
<u>TITLE</u>	<u>CODE</u>	<u>TITLE</u>	<u>CODE</u>
TOTAL COST CENTER	TOT	PROFIT	PROFIT
PRODUCT GROUP A	GR-A	REVENUE	REV
PRODUCT A1	A1	UNITS PRODUCED	UNITS
PRODUCT A2	A2	UNIT COST	UC
PRODUCT A3	A3	TOTAL COST	COST
PRODUCT GROUP B	GR-B	LABOR	LABOR
PRODUCT B1	B1	HOURS	HOURS
PRODUCT B2	B2	LABOR RATE	LR
PRODUCT B3	B3	OVERHEAD	OH
		OVERHEAD RATE	OR
		MATERIAL	

There will be 99 combinations of product and financial data and useful time-oriented information that could exist for each of them.

	PROFIT	REVENUE	UNITS PRODUCED	UNIT PRICE	TOTAL COST	LABOR COST	HOURS	LABOR RATE	OVERHEAD	O.H. RATE	MATERIAL
TOTAL											
PRODUCT GR A											
PRODUCT A1											
PRODUCT A2											
PRODUCT A3											
PRODUCT GR B											
PRODUCT B1											
PRODUCT B2											
PRODUCT B3											

FIGURE 3

Using the code defined previously, the total profit could be retrieved in the following way:

```
FF CC PROFIT:      (FINANCIAL FACTOR -- PROFIT WITHIN
PRODUCT TOT:      COST CENTER CC)
GET P TI DI:
```

The 'TI' identifies our interest in timespread data and the 'DI' option would produce the following titled display:

```
FIN FACTOR  PROFIT
PRODUCT      TOTAL
```

P

MO/YR (MONTHS)

1/74	195.	260.	325.	396.	467.	514.
7/74	532.	485.	372.	313.	307.	343.
1/75	.	.	.	.	.	.

All retrievals produce a new data set (in this case, 'P') which can be displayed, graphed, or used in a variety of computations.

#### Data Base Input

New information can also be created through the insertion of data into hierarchial EIS data bases. In the above matrix of 99 cells only 36 contain basic information (the non-shaded areas within Figure 3). The other 63 combinations can be created through a series of pre-defined arithmetic operations which are executed when and only when new information requires it. To illustrate the process, consider the effect of a change in the number of hours required to produce Product B2. Fourteen of the 99 elements are effected by that one change (profit, total cost, total labor cost, hours and overhead for total cost, product group B and product B2). This illustrates the problem often experienced in planning, budgeting and estimating operations when assumptions change or alternatives are being explored. The problems of 'keeping track' grow rapidly as the size, inter-relationships and number of dimensions increase. Unfortunately, what often happens in the absence of effective computer resources is that problems are 'cut down' to a size which an army of persons with desk calculators can support. The results are almost always less than satisfactory.

The EIS System supports not only retrieval, but also accepts on-line or off-line entry into these same structured data bases. The entry process is the exact reverse of data retrieval. First, the appropriate elements of each dimension are defined, then the new information is 'assembled' into the file.

Finally, the system automatically develops and inserts whatever additional information the new data and the data base logic require.

In summary, users have a great deal of flexibility in defining EIS data bases. They may be multi-dimensional collections whose elements are defined, named, and related as the application requires. In most cases, the work to set up a customized data base can be accomplished in a matter of days. For that reason, these data bases are used in a wide variety of business, government, managerial and technical applications where time, unit, or cum data are required.

### INFORMATION DISPLAY

The simplest form of display within EIS is handled by the DISPLAY command. In the example below, the display for X1, HRS and COST would produce the following result. (The first two of these were created through the INPUT command and the third was developed through a COMPUTE instruction.)

DISPLAY X1 HRS COST /2/:

X1

22.00	20.00	33.00	31.00	25.00	24.00	21.00
22.00	25.00	29.00	33.00			

HRS

MO/YR (MONTHS)

1/75	33.00	44.00	55.00	67.00	79.00	87.00
7/75	90.00	82.00	63.00	53.00	52.00	58.00
1/76	64.00	68.00	70.00	66.00	55.00	41.00
7/76	33.00	29.00	27.00	26.00	25.00	24.00

COST

MO/YR (MONTHS)

1/75	195.03	260.04	325.05	395.97	466.89	514.17
7/75	531.90	484.62	372.33	313.23	307.32	342.78
1/76	414.08	439.96	452.90	427.02	355.85	265.27
7/76	213.51	187.63	174.69	168.22	161.75	155.28

The number between the slashes defines the decimal significance requirement. Additional options allow the user to restrict output to selected time or units.

### Form Processor

The EIS Form Processor gives users full control over printed output at the terminal or on a high-speed printer. Report generation is greatly simplified by a number of special features which have been incorporated in the three form commands. These include automatic time and date, word centering, optional bracketing of negative numbers, automatic month and year titling, internal sub-totaling, modification of time interval definitions and inclusion of graphics. Most importantly, it was written so it can be used easily by non-computer professionals, and it eliminates the constraint of 'standard reports'. Examples of output produced through the Form Processor are included with graphs in Figures 14 through 19.

### Graphics

Computer graphics has been one of the glamorous subjects of data processing for many years, but to date, graphics has failed to make the impact on business decision making that is needed. Until recently, the cost of equipment has been a major deterrent to wide spread use, but cost is no longer a big factor. The real problem is that the potential of business graphics has been 'contained' within the computing fraternity. Much of the work in graphics has been an outlet for the creative instincts of systems programmers as an end in itself. Moreover, the development of graphics has generally required a programmer capable of combining the necessary subroutines from complex graphics library.

Graphics, like all of the other capabilities within EIS, was developed for use by persons in business and management, not computing. For that reason, nearly all EIS graphs can be created by a single, one-line instruction. (Throughout this section, the command used to produce each graph will be superimposed on each example. All graphs are shown approximately one-third scale.) The command GRAPH HRS will result in a line drawing of data set 'HRS' in which the scaling,

GRAPH HRS

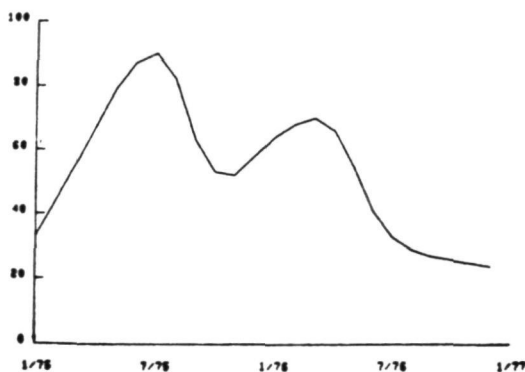


FIGURE 4

GRAPH ABC

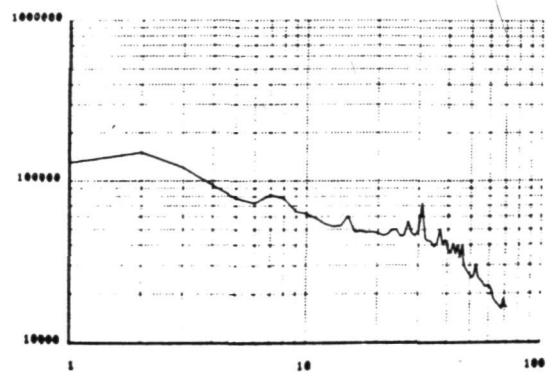


FIGURE 5

line type, graph, and character size, and selection of graphic form take place automatically (see Figure 4). In this example, the system selects a linear scale with time defined on the horizontal axis because timespread data is usually presented in that way. The unit costs of manufactured goods are often related to one another exponentially so the system defaults the display of unit data to a logarithmic scale (see Figure 5). In both cases, the selection of a graphic form is made possible because the data itself is identified as being time or unit oriented.

Cum data is presented in a variety of ways depending on the nature of the information and the audience for which it is being prepared. The standard response is a series of connected points drawn on a standard time grid (see Figure 6). The only difference between this and a time graph is that the horizontal axis is labeled with the sequence number of positions for a cum data set.

GRAPH X1

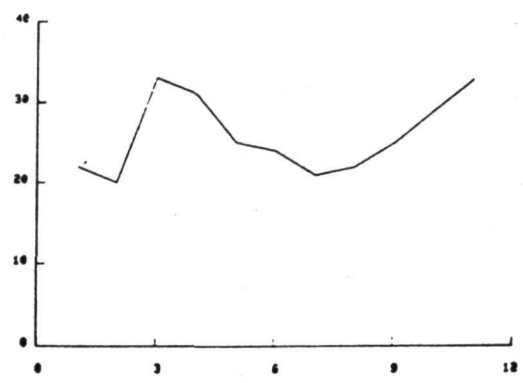


FIGURE 6

GRAPH X1 /PIE/

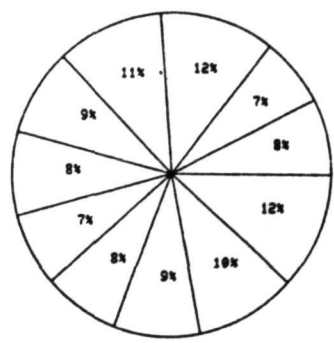


FIGURE 7

GRAPH X1 /HB/

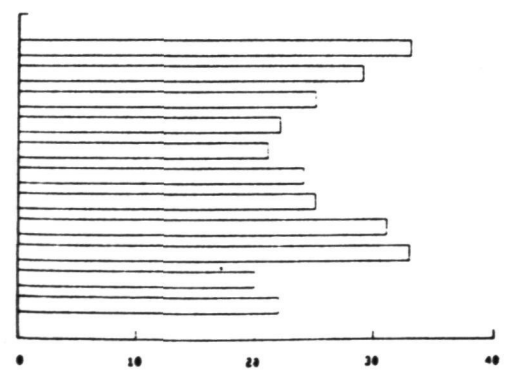


FIGURE 8

GRAPH X1 /VB/

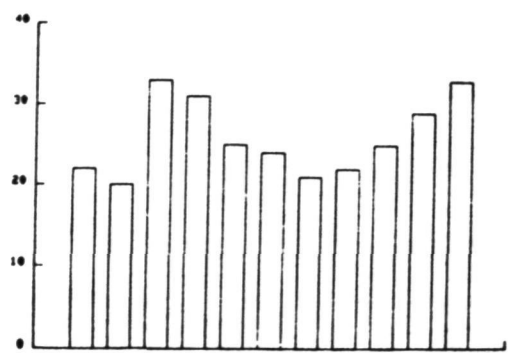


FIGURE 9



The same cum data is displayed in Figures 7 through 9. In each case, a different option is exercised. The 'PIE' in Figure 7 causes data set X1 to be represented by a pie chart. The numbers in each sector represent that sector's percent of the whole. An additional option would supervise the placement of titles in each of these sections in addition to the numeric values. Figures 8 and 9 are produced through the horizontal and vertical bar options.

A large number of options are available with which special graphic effects can be obtained. These include the selection of graphic form, line control, shading, placement on the page, multiple images on a page, control of scaling and labeling, plotting of cum data sets, optional grid lines and stack charts. Most of these are activated through a two-character abbreviation. Various possibilities are illustrated in Figures 10 through 13. Three data sets of five elements each are graphed in a comparative bar chart shown in Figure 10.

GRAPH B1 B2 B3 /VB/

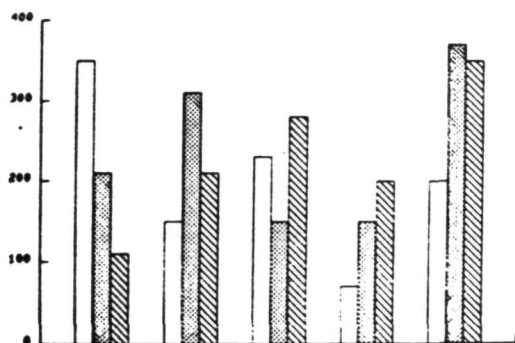


FIGURE 10

GRAPH A1 A2 A3

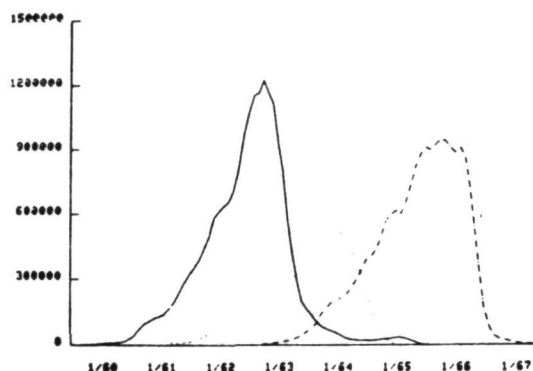


FIGURE 11

GRAPH B /BAR TX (ML) CUM/

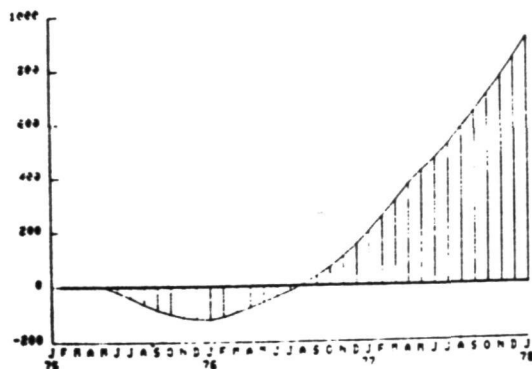


FIGURE 12

GRAPH HRS /BAR/ HRS /GR TX (M QL)/  
HRS/GR CUM TX (Q QL)/HRS/BA SQ SE/

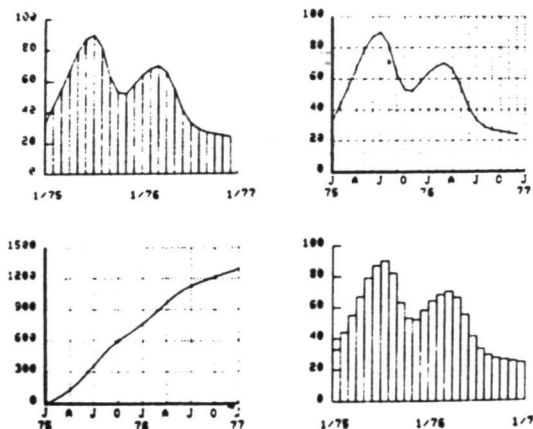


FIGURE 13

The VB option is used to position the respective elements of data sets, B1, B2 and B3, next to one another in a vertical bar chart. The shading is taken care of automatically. Similarly, the system automatically selects a different line type for each of the timespread data sets shown in Figure 11.

The cum and bar options referenced in Figure 12 automatically computes and plots the cum of cash flow line CFL and shades in the space between the curve and the zero reference line. The TX(ML) option in this same example causes the time axis to be labeled for each month.

The SE option was used to produce the separate charts shown in Figure 13. Otherwise, the data sets (all of which are the same) would have been plotted in the same space. The other options illustrate, again, the variety of ways in which the same information can be presented.

Charts are used in a variety of ways, sometimes for quick reference to data and other times, as a part of formal reports. In the latter case, the graph command is usually used in combination with alpha-numeric information produced by the form processor. The graphics are produced in exactly the same way as shown above except they are reduced in size to permit the addition of alpha-numeric information. Since the alpha-numeric data produced by the form processor are related to column and row, graphic grid positioning is defined in the same way. The graph in Figure 14 was bounded within columns 12 and 60, and rows 15 and 30. This information is included in the graph command between parentheses (e.g., GRAPH (15 30 12 60) B1 B2 B3.

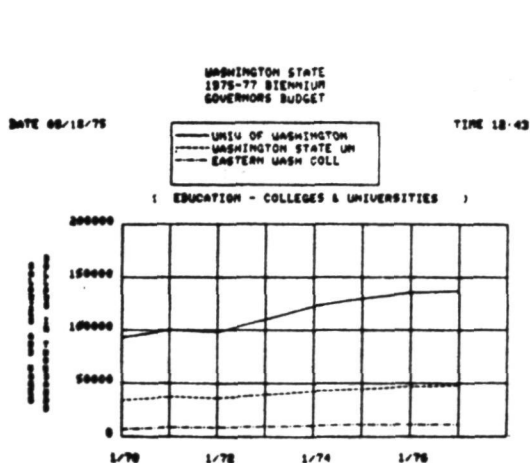


FIGURE 14

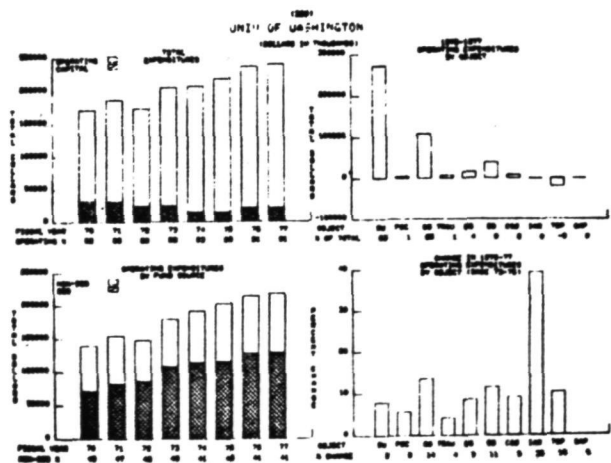


FIGURE 15

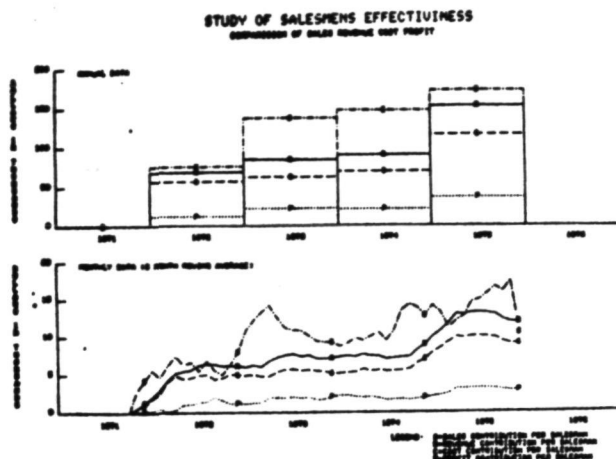


FIGURE 16

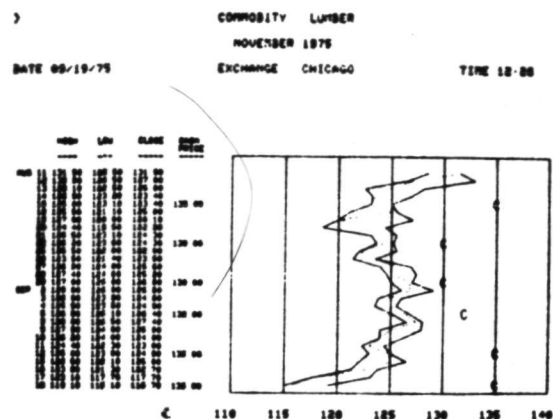


FIGURE 17

The examples shown in Figures 14 through 17 are representative of the form processor/graphic output from the various budget, performance measurement and analysis applications written in the EIS language. Even in these more formal charts, the graphic portion is almost always completed with a single graphics command of one or two lines.

EIS graphics are available on-line through Tektronics 4010 or 4014 terminals or off-line on the Stromberg Carlson 4020 system. The off-line service extends graphics capability to persons without convenient access to graphic terminals and provides an economical alternative for large volume work. In either case, the graphs are produced with the same commands and include all of the same options.

## EIS APPLICATIONS

### Interactive Use

New users are introduced to the system in three-day classes during which they learn to use each EIS instruction by typing them one at a time at the computer terminal. The system interprets the instruction and either performs the task or responds with a brief, but clear diagnostic.

A great number of EIS services are initiated, one instruction at a time, because in many cases, each step may depend on the result of a previous instruction. Each EIS command can accomplish a great deal, however. With one simple COMPUTE instruction, for example, a user can multiply a time-phased estimate of hours by day, month, quarter, etc., by a labor rate which also varies by time period. The answer can be available by period, in total or cummed forward or backward in time, through that one instruction. The special functions provide an even greater service. With a single command, a user can compute the return on investment on a data set of unlimited length or extend the assumption of a complex learning curve over hundreds of units.

Data retrieval is heavily used by persons engaged in cost research and managers comparing expenditure against plans. Interactive use of the EIS System also provides new users with a good vehicle to develop and extend their EIS skills.

### EIS Programs

The largest percent of EIS applications are tasks which are performed many times--often with slight variations from one use to the next. These are customarily entered into a program file and stored on the computer disk under a name assigned by the user. The obvious advantage of pre-programmed files is that they can be executed more quickly and for less cost than equivalent services requested from the terminal. In addition, programmed files are available to several other persons. In either case, the commands are expressed in an identical way.

Many of the programmed files contain a small number of instructions. The output shown in Figures 18 and 19 are examples of this process. The learning curve program is produced by instructions in a file named LEARNING CURVE. The 11 commands supervise the retrieval of the requested unit cost data for airplane XYZ, determine the best fit curves for units 1 through 40, and units 41 through 70, and display the results with companion form generator file which was also written by the user with the EIS language (see Figure 18). The service is initiated by the CALL instruction. The four arguments

CALL	LEARNING CURVE	PROGRAM	COST	WBS	BREAK
		NAME	ELEMENT		POINT

define the program name, organization, hardware component and breakpoint at which different learning curves are to be developed. This simple example combines the

use of computation, the graphics and form processor and retrieval from a structured data base. The arguments which may be substituted in this example allow this file to be used for thousands of combinations.

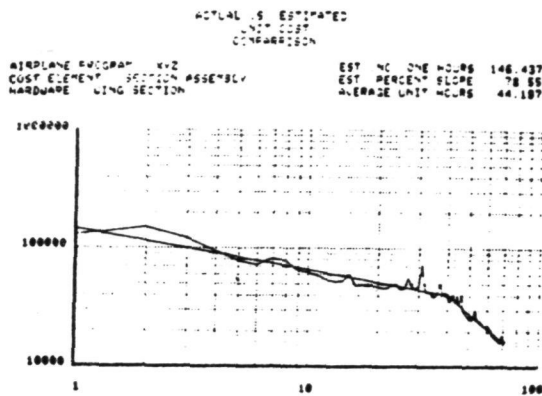


FIGURE 18

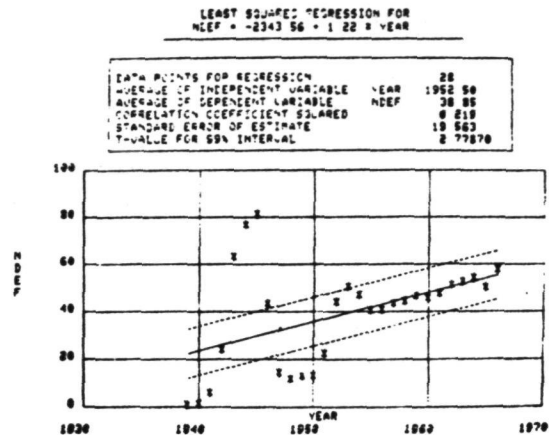


FIGURE 19

The output shown in Figure 19 is provided by a small file which accepts two variables for which it performs a linear regression to determine a least-squares fit. All that is required to initiate the process is to call file 'LINEAR REGRESSION' and identify the two variables and a confidence boundary. If transformations are required (e.g., changing the values to current dollars) they can be performed easily through a COMPUTE command before the regression program is called.

Some of the applications require several thousand instructions. One such application is a complex airplane estimating model that was developed by a cost research group of Boeing Airplane Company. The extensive and continuing cost research has been accelerated by the interactive active environment of the system. The availability of specialized commands and large historic data bases eliminated much of the work which would otherwise have been required.

### Examples

Without any attempt to be complete, the following are examples of areas in which the EIS System is being used.

Research. - One of the major objectives of EIS was to promote cost research. Large data bases provide ready access to organized collections of information. The special commands and high-level language simplify user access to computing resources. The time-sharing environment and interactive graphics accelerate the discovery of fact.

Estimating Services. - Major (and minor) proposals involving hundreds of hardware components, numerous cost centers and multi-time periods are developed within the system. The cost of service is low because the interactive environment, structured data base and user involvement are ideally suited to estimating requirements.

Planning, Budgeting, and Performance Measurement. - EIS is the vehicle for planning and performance measurement for many organizations. The Washington State Legislature, for example, used EIS as the principle resource for analyzing, comparing and constructing the six billion dollar state budget. The expenditure history and recommendations for 600 agencies and programs were stored in EIS data bases. Alternate proposals were introduced into the system from staff offices, committee hearing rooms and off the floor of the Senate and House. The impact at all levels of government was known immediately. The near-term availability of monthly expenditures from each agency will allow the same system to be used to monitor performance.

National Computer Hook-Up. - Boeing Computer Services uses EIS to communicate information between its separate divisions which are located throughout the United States. Plans and monthly actuals for a large number of product lines are entered into the system at each divisional location. The data is automatically summarized for use at both the remote and headquarters locations. Besides the usual benefit, the application shortens the time spent in financial consolidation by several days. Communication between cities is a no-cost benefit of operating on a national time-sharing network.

Negotiations. - EIS programs are used to negotiate prices and overhead rates with customers and subcontractors. A versatile set of support programs can usually accomplish more than a large group of people. Negotiators at work any place in the country can still be in touch with the home office and large data bases through the system.

In every case, users created the capability or played a big part in their development.

## SUMMARY

Extending the capabilities of decision makers and their staffs is one of the great challenges for management and computing alike. As a class of problems, managerial decisions have an enormous impact on the profit and future prospects of an organization. Unfortunately, all too often, those choices must be made within impossible time constraints and in the absence of sufficient information. Historically, because of the need for quick response, the continuing variability of requirements over time and the necessity for managerial involvement, the contribution of computing has been limited.

For more than five years, hundreds of persons within the traditional business functions have used EIS to solve their own problems. As a result, the computer

is now an accepted part of many activities previously regarded as too specialized for computer service. There is every reason to expect this process to continue. Toward that end, EIS is continually being extended to respond to the growing opportunity for service to management.

# SURFACE FITTING THREE-DIMENSIONAL BODIES

Fred R. DeJarnette and C. Phillip Ford III

North Carolina State University

## SUMMARY

The geometry of general three-dimensional bodies is generated from coordinates of points in several cross sections. Since these points may not be smooth, they are divided into segments and general conic sections are curve fit in a least-squares sense to each segment of a cross section. The conic sections are then blended in the longitudinal direction through longitudinal curves which may be used to define the conic sections in the cross-sectional planes. Both the cross-sectional and longitudinal curves may be modified by specifying particular segments as straight lines or specifying slopes at selected points. Slopes may be continuous or discontinuous and finite or infinite.

This method was used to surface fit a  $70^\circ$  slab delta wing and the HL-10 Lifting Body. The results for the delta wing were very close to the exact geometry. Although there is no exact solution for the lifting body, the surface fit generated a smooth surface with cross-sectional planes very close to prescribed coordinate points.

## INTRODUCTION

Many disciplines require a mathematical description of three-dimensional surfaces which cannot be represented by simple mathematical expressions. The location and slope of points on a body are needed in the analysis of structures and inviscid flow fields. Viscous flow-field analyses require the body curvature in addition to location and slopes. The geometrical properties of simple shapes like spheres, cones, ellipsoids, and paraboloids can be described by relatively simple mathematical equations. However, many configurations of interest today, such as the space shuttle, are complex three-dimensional shapes whose geometry cannot be described very easily. In many cases all the information that is known is a drawing with a plan view, a side view, and several cross sections of the body. Sometimes models of a vehicle are available and coordinate positions can be accurately measured on them. On the other hand, slopes cannot be accurately measured on a drawing or model, and the measured radii of curvature are even less accurate. This report develops a method for surface fitting mathematical relations to complex three-dimensional bodies. The method yields accurate coordinates and slopes and reasonably accurate radii of curvature at any position on the body.

Previous approaches to surface fitting three-dimensional bodies generally divided the surface into "patches" and represented each patch by flat surfaces, cubic or higher order polynomials, or conic sections (e.g. see refs. 1-5).



Each of these methods has undesirable features in the forms used previously. Flat surfaces are completely unacceptable if radii of curvature or continuous slopes are needed. Cubic or higher order polynomials often lead to unwanted wiggles and bulges since they allow points of inflection. In ref. 1, DeJarnette applied the method of double splines, which used bi-cubic interpolation, to surface fit the coordinates of points in several cross-sectional planes of three-dimensional bodies. However, it was found that bulges and/or dimples occurred in these surfaces, particularly when the thickness was much smaller than the span.

Reference 2 approximated the shape of a space shuttle with elliptical cross sections with different ellipticity on the windward and leeward sides. Cubic polynomials were used to define segments of the plan and thickness distributions. The coefficients of these polynomials were chosen to make the slopes continuous across boundaries of the longitudinal segments. However, points of inflections were found to occur inside the segments and thus gave undesirable bumps and wiggles, which in turn significantly affected the surface pressure and heating distributions calculated with this geometry.

For sometime conic sections have been used to describe segments of aircraft contours (ref. 3). Both longitudinal and transverse contours were represented by conic sections, but the slopes at the ends of each segment had to be measured. As mentioned earlier, slopes are difficult to measure accurately. On the other hand, conic sections cannot have inflection points, and this feature is useful in eliminating undesirable wiggles.

Coons (ref. 4) developed a technique to describe three-dimensional surfaces by using blending functions to blend the surface between the boundary curves of each patch. A major difficulty of applying Coons' method is that the user must supply the coordinates, slopes, and twists (cross derivatives) at all four corners of each patch. This information is generally difficult to determine, particularly the cross derivatives (See ref. 5). Craidon (ref. 6) used this method with the twists arbitrarily chosen to be zero to surface fit three-dimensional geometries.

The method presented here uses data points in cross-sectional planes to curve fit segments of general conic sections in a least-squares sense. The conic sections are then blended in the longitudinal direction by use of coordinate points which define the conic sections in the cross-sectional planes. This technique has the advantage of allowing the user to continually modify the cross-sectional curves and the longitudinal curves until the body shape has the desired features. Also, discontinuous slopes in both the circumferential and longitudinal directions may be specified.

#### SYMBOLS

$a_{m,i}, b_{m,j}$  parameters defined by Eqs. (B5) - (B16)

$A_1$  thru  $A_5$  coefficients of conic section given by Eq. (18) in global coordinates

$A_j, B_j, C_j$ $D_j, E_j, F_j$	coefficients of conic section given by Eq. (1) in local coordinates
$d_p$	constant vector in Eq. (19)
$G_{pq}$	matrix used in Eq. (19)
$K_j$	data point number of first control point in segment $j$
$m_j, n_j$	slopes defined by Eqs. (5) and (6)
$N$	number of segments in a cross section
$P_j, Q_j$	terms defined by Eqs. (A2) and (A3)
$r_{j,k}$	residual in Eqs. (B1) and (B2)
$R_j$	term defined by Eq. (A5)
$\bar{x}, \bar{y}, \bar{z}$	Cartesian coordinates, see figure 2
$y, z$	local coordinates, see figure 3
$\alpha_j, \beta_j, \gamma_j$	coefficients defined by Eqs. (15), (16), and (17)
$\theta_j$	slope of segment $j$ , see figure 5
$\Delta\theta_j$	$\theta_j - \theta_{j-1}$
Subscripts:	
$h$	intermediate point in a cross-sectional segment
$j$	segment number in a cross section
$k$	data point number in a cross section

## ANALYSIS

In a typical application, the geometry of a body must be determined from a model or a three-view drawing. Generally several cross sections are obtainable from the model or drawing, and the coordinates of data points on the boundary of these cross sections can be measured or calculated. A three-dimensional

surface must then be fit through all the cross sections. In the method presented here, the data points in each cross section are divided into segments and general conic sections are fit to the data points in each segment as shown in figure 1. A three-dimensional surface is then generated by "blending" the cross-sectional curves in the longitudinal direction. Consider first the curve-fitting of data points in a cross-sectional plane.

#### Curve-Fit in a Cross-Sectional Plane

The data points in a cross-sectional plane are generally not completely smooth, and in these cases a smooth curve cannot be made to pass through every data point. Therefore, the data points are divided into segments, and a general conic section is fit in a least-squares sense through the data points in that segment. The conic section is constrained to go through the control points (end points of a segment) and also have a continuous slope at each control point unless otherwise specified (See figure 1).

A three-dimensional coordinate system  $\bar{x}, \bar{y}, \bar{z}$  is used with  $\bar{x}$  in the longitudinal direction and  $\bar{x} = \text{constant}$  is a cross-sectional plane (See figure 2). Let  $j$  denote the segment number in a cross-sectional plane as shown in figure 1, with the first segment starting on the positive  $\bar{y}$  - axis and  $j$  increasing clockwise. It is convenient to use a local coordinate system  $y, z$  for each segment with the origin on the first control point and the positive  $y$  - axis passing through the control point at the other end of the segment (See figure 3). In this local coordinate system it is easy to investigate possibilities of complex roots and interpret the coefficients of a general conic section geometrically, whereas it is difficult to interpret them geometrically in the global coordinates  $\bar{y}, \bar{z}$ . It is assumed that the local coordinate  $z$  is single-valued in each segment.

The equation for a general conic section for the  $j^{\text{th}}$  segment is given by ref. 7 as

$$A_j y^2 + B_j yz + C_j z^2 + D_j y + E_j z + F_j = 0. \quad (1)$$

Only five of the six coefficients are independent since the equation may be divided by any non-zero coefficient. The constraints that the conic sections pass through the two control points

$$y = 0, z = 0 \quad \text{and} \quad y = y_j, \quad z = 0 \quad \text{yields}$$

$$F_j = 0 \quad (2)$$

$$D_j = -A_j y_j \quad (3)$$

By differentiating Eq. (1) with respect to  $y$ , the slope in the local coordinates is given by

$$\frac{dz}{dy} = \frac{A_j y_j - 2A_j y - B_j z}{B_j y + 2C_j z + E_j} \quad (4)$$

The slopes at the control points are generally not known, so define

$$m_j = \left( \frac{dz}{dy} \right) \text{ at } y = 0, z = 0 \quad (5)$$

and

$$n_j = \left( \frac{dz}{dy} \right) \text{ at } y = y_j, z = 0. \quad (6)$$

By using these two equations in Eq. (4) it follows that

$$E_j = A_j y_j / m_j \quad (7)$$

and

$$B_j = -A_j \left( \frac{1}{m_j} + \frac{1}{n_j} \right). \quad (8)$$

These equations show that the end slopes do not affect the coefficient  $C_j$ . On the other hand, the product  $A_j C_j$  determines the nature of the general conic section (ref. 7). If the discriminant  $(B_j^2 - 4A_j C_j) = 0$  the conic is a parabola, if the discriminant  $< 0$  the conic is an ellipse, and if the discriminant  $> 0$  the conic is hyperbola. Of particular interest is the possibility of complex roots when solving for  $z$  as a function of  $y$  in the region of interest. It is shown in Appendix A that  $z$  will not be complex in the region  $0 \leq y \leq y_j$  if

$$A_j C_j \geq \left( A_j / m_j \right) \left( A_j / n_j \right). \quad (9)$$

For prescribed slopes  $m_j, n_j$  figure 4 shows how the product  $A_j C_j$  affects a conic section.

Unless a slope is specified at a control point, the method used here constrains the slope in global coordinates  $(\bar{y}, \bar{z})$  to be continuous at a control point. This makes the conic section in one segment dependent on data points in other segments as well as its own. From figure 5 it can be seen that continuity of slope at control point  $j$  requires

$$\tan^{-1} n_{j-1} = \tan^{-1} m_j + \Delta\theta_j. \quad (10)$$

This equation can be expanded and rearranged into the form

$$\frac{A_{j-1}}{(A/n)_{j-1}} = \frac{(A/m)_j \sin \Delta\theta_j + A_j \cos \Delta\theta_j}{(A/m)_j \cos \Delta\theta_j - A_j \sin \Delta\theta_j}. \quad (11)$$

Because this last equation is non-linear in the coefficients  $A_j$ , it would cause difficulty in obtaining a solution for them. However, as mentioned previously only 5 of 6 coefficients in a given segment are independent; therefore, an additional constraint can be imposed without affecting the resulting equation for the general conic section. The additional constraint used is to equate separately the numerators and denominators of both sides of Eq. (11). This procedure yields two equations, both linear in the coefficient  $A_j$ , which may be rearranged to give

$$(A/n)_j = \left[ A_j \cos \Delta\theta_{j+1} - A_{j+1} \right] / \sin \Delta\theta_{j+1} \quad (12)$$

and

$$(A/m)_j = \left[ A_{j-1} - A_j \cos \Delta\theta_j \right] / \sin \Delta\theta_j. \quad (13)$$

For the first segment ( $j = 1$ ),  $m_1$  is specified and Eq. (13) is not needed, and also for the last segment ( $j = N$ ),  $n_N$  is specified and Eq. (12) is not needed.

Now substitute Eqs. (2), (3), (7), (8), (12), and (13) into Eq. (1) to obtain the equation for the general conic sections as

$$\alpha_j A_{j-1} + \beta_j A_j + \gamma_j A_{j+1} + z^2 C_j = 0 \quad (14)$$

where

$$\alpha_j = \begin{cases} \left[ zy_j - zy \right] / \sin \Delta\theta_j & \text{for } j > 1 \\ 0 & \text{for } j = 1 \end{cases} \quad (15)$$

$$\beta_j = \begin{cases} y^2 - (1/m_1 + \cot \Delta\theta_2) yz + y_j z / m_1 - y_j y & \text{for } j = 1 \\ y^2 + (\cot \Delta\theta_j - \cot \Delta\theta_{j+1}) yz - \cot \Delta\theta_j y_j z - y_j y & \text{for } 1 < j < N \\ y^2 + (\cot \Delta\theta_N - 1/n_N) yz - \cot \Delta\theta_N y_N z - y_N y & \text{for } j = N \end{cases} \quad (16)$$

$$\gamma_j = \begin{cases} yz / \sin \Delta\theta_{j+1} & \text{for } j < N \\ 0 & \text{for } j = N \end{cases} \quad (17)$$

These equations indicate that there are only two unknowns in each segment,  $A_j$  and  $C_j$ . However, there is one more segment than interior control points which means that one of these coefficients is arbitrary. Here  $A_1 = 1$  is used arbitrarily (unless the conic section requires  $A_1 = 0$ ) and the independent unknowns become  $C_1, C_j, A_j (j = 2, \dots, N)$ . If there were as many data points as unknowns, and if some conic section could be made to pass through these data

points, then Eq. (14) could be applied to all the data points to give  $(2N-1)$  linear equations for the  $(2N-1)$  coefficients  $C_1, C_j, A_j$  for  $j = 2, \dots, N$ . However, there are generally more than  $(2N-1)$  data points, and if Eq. (14) were applied to all of them an overdetermined system of linear equations (See ref. 8) would result. Therefore, a least-squares solution of the overdetermined system is used to determine the coefficients  $C_1, C_j, A_j$  for  $j = 2, \dots, N$ . This procedure is described in Appendix B. After obtaining the coefficients in this manner, the inequality of Eq. (9) is checked for the possibility of complex roots for  $z$  in each segment. If the inequality of Eq. (9) is not satisfied, the coefficient  $C_j$  in that section is replaced by the value obtained using the equality sign in Eq. (9). As mentioned earlier, the coefficient  $C_j$  does not affect the slopes of the curve at the end points of the segment, and hence  $C_j$  does not affect other segments. The equality sign in Eq. (9) gives a hyperbola of 2 straight lines as shown in figure 4.

In order to solve for  $z$  as a function of  $y$  from Eq. (14), a quadratic equation must be solved, and the proper choice of the + or - sign must be determined beforehand for each segment. It is shown in Appendix A that in order to make  $z = 0$  at  $y = 0$  (a control point at the beginning of segment  $j$ ), the + sign must be used if  $(A/m)_j > 0$  and the - sign must be used if  $(A/m)_j < 0$ .

Once the coefficients  $A_j$  and  $C_j$  are determined, all the conic sections for that cross section are completely defined. In order to put these results into a form suitable for "blending" the cross sections in the longitudinal direction, the conic sections for each segment are redefined in terms of 4 points: the two control points at the end of the segment, a slope point which determines the slope at the end points, and finally an intermediate point on the curve between the end points (See figure 6). The 3 points on the curve and the two slopes at the end points of a segment are sufficient to determine new coefficients  $A_1, A_2, A_3, A_4$ , and  $A_5$  for the general conic section\*,

$$A_1 \bar{y}^2 + A_2 \bar{y}\bar{z} + A_3 \bar{z}^2 + A_4 \bar{y} + A_5 \bar{z} + 1 = 0 \quad (18)$$

in global coordinates  $\bar{y}, \bar{z}$ . This process is applied to each segment in a cross section, and therefore the 5 coefficients become functions of the longitudinal coordinate  $\bar{x}$  when the segments of a cross section are blended with corresponding segments in the other cross sections.

#### Longitudinal Variation of Cross Sections

In order to determine the longitudinal variation of the coefficients in Eq. (18), a three-dimensional curve is fit through each of the 4 points used to define the conic section of corresponding segments (See figure 7). In contrast to the cross-sectional data points, these curves must pass through each of the points in the longitudinal direction. They are represented by their projections

---

\* Note that the conic section given by Eq. (18) must have the constant 1 replaced by 0 if the curve is to pass through the origin.

in the  $\bar{x}, \bar{y}$  and  $\bar{x}, \bar{z}$  planes; hence, two planar curves are used to represent each three-dimensional curve. The parametric method of cubic splines (ref. 9) is used to curve-fit each planar curve, with the chordal distance between the coordinate points as the parameter. The parametric spline allows infinite slopes whereas the regular spline does not.

This method worked very well for some test cases but had trouble with others because of the bumps and wiggles inherent to cubic splines. These bumps and wiggles can be corrected by specifying slopes in the longitudinal direction at certain cross sections. Also parts of the longitudinal curves can be specified as straight lines.

Consider now the longitudinal variation of a conic section. In each cross-sectional plane, Eq. (18) will hold but the coefficients  $A_1, A_2, A_3, A_4$ , and  $A_5$  will vary with  $\bar{x}$ . As mentioned previously, these coefficients are determined by 4 defining points (the two control points, an intermediate point, and the slope point). For each segment, the 5 equations used to determine the coefficients  $A_q$  ( $q = 1, \dots, 5$ ) are formed by applying Eq. (18) to the 3 points on the cross-sectional curve (the two control points  $\bar{y}_0, \bar{z}_0$  and  $\bar{y}_1, \bar{z}_1$ , the intermediate point  $\bar{y}_h, \bar{z}_h$ ) and the slopes at the ends of the segment using the slope point  $\bar{y}_s, \bar{z}_s$  (See figure 6). This procedure yields the following 5 equations:

$$\sum_{q=1}^5 G_{pq} A_q = d_p \quad p = 1, \dots, 5. \quad (19)$$

At any longitudinal position, Eq. (19) can be solved by any standard matrix inversion routine to determine the coefficients  $A_q$ . The derivatives  $dA_q/d\bar{x}$  and  $d^2A_q/d\bar{x}^2$  can be obtained by differentiating Eq. (19) and successively solving the resulting system of linear equations. The elements of  $G_{pq}$  and their derivatives with respect to  $\bar{x}$  are obtained from the three-dimensional curves which were spline-fit through the 4 points used to define the conic section for that segment in each cross-sectional plane.

#### COMPUTATIONAL ALGORITHM

Given the set of data points  $(y_k, z_k)$  in cross-sectional planes at several longitudinal stations,

- (1) For each cross-sectional plane, divide the data points into segments so that a conic section can be curve-fit to the data points in each segment by the least-squares technique developed herein.
- (2) If the curves fit to the cross-sectional data points are not satisfactory, modify them by one or more of the following methods: (a) define new boundaries (control points) for segments, (b) specify slope(s) at control point(s) (slopes may be finite or infinite and continuous or discontinuous), (c) specify selected segments as straight lines, (d) a specific conic section can be

specified for a segment by prescribing the slopes at the ends of the segment and using only one datum point between the end control points.

- (3) Represent the conic section for each segment in a cross-sectional plane in terms of the two control points at the ends of the segment, an intermediate point, and the slope point (See figure 6).
- (4) For each point found in step (3), spline-fit a three-dimensional longitudinal curve through it and the corresponding points in other cross-sectional planes (See figure 7).
- (5) If the longitudinal curves are not satisfactory, modify them by one or more of the following methods: (a) specify slope(s) at longitudinal station(s) (slopes may be finite or infinite and continuous or discontinuous), (b) specify selected longitudinal segments as straight lines, (c) redefine the boundaries (control points) of the segments in the cross-sectional planes so that the points used for the longitudinal spline-fit form a smooth curve.
- (6) The geometrical properties of the surface may be computed in polar or Cartesian coordinates at any position  $\bar{x}$ ,  $\bar{y}$  by the following steps:
  - a. Locate  $\bar{x}$  between two consecutive longitudinal stations, and then locate the cross-sectional segment which contains  $\bar{y}$ .
  - b. Use the spline function to calculate the coordinates, slopes, and second derivatives of the 4 longitudinal curves for this segment at  $\bar{x}$  (See figure 7).
  - c. Calculate the coefficients  $A_q$  ( $q = 1, \dots, 5$ ) of the conic section at this location by use of Eq. (19). Determine the first and second derivatives of  $A_q$  with respect to  $\bar{x}$  from the first and second derivatives of Eq. (19).
  - d. Calculate the body position  $\bar{z}$  from Eq. (18), and the derivatives of  $\bar{z}$  with respect to  $\bar{x}$  and  $\bar{y}$  from derivatives of Eq. (18).
- (7) After a satisfactory surface fit has been obtained, the data which must be retained for a geometry subroutine package are the coordinates and longitudinal slopes of the longitudinal curves at those longitudinal stations where cross-sectional data points were given. Then, the geometrical properties of the surface can be calculated at any position by the method outlined in step (6) above.



## APPLICATION TO 70° DELTA WING

The surface fitting method was applied to the 70° slab delta wing shown in figure 8. This example was chosen because it illustrates many of the options available to modify the longitudinal curves and because the results can be compared with an exact solution.

Cross-sectional data was used as shown in figure 8, and due to symmetry only the first quadrant is used. Two segments (three control points) are needed to represent the cross section of  $\bar{x} = 10$ . The first segment is a straight line and the second is one-fourth of an ellipse. The least-squares curve-fit technique represents the ellipse exactly by specifying a zero slope at control point  $j = 2$ , an infinite slope at control point  $j = 3$ , and one datum point between these two control points. Although two segments must also be used for the other two cross sections, only one segment is necessary to specify the circle at  $\bar{x} = 0.65798$  and the ellipse at  $\bar{x} = 1.0$ . Therefore, the first two data points, which are also control points, are made coincident. Then the exact curves are calculated from the least-squares curve-fit by specifying a zero slope at control point  $j = 2$ , an infinite slope at control point  $j = 3$ , and one datum point between these two control points. The exact location of this datum point is irrelevant except that it must lie on the desired curve.

The three-dimensional longitudinal curve is represented by its projections in the  $\bar{x}, \bar{y}$  and  $\bar{x}, \bar{z}$  planes. Therefore, 16 longitudinal planar curves are used for this example (8 for each cross-sectional segment). Modifications were made to the initial spline-fits to 14 of these curves since exact slopes are readily obtained from figure 8.

The geometrical properties and their derivatives were calculated at four circumferential positions for  $\bar{x} = 0.3, 1.0, 2.0$ , and  $5$ . The results at  $\bar{x} = 1.0, 2.0$ , and  $5.0$  are exact (within the accuracy of single precision on the IBM 360/175 computer), whereas some inaccuracies were noted at  $\bar{x} = 0.3$ .

## APPLICATION TO HL-10 LIFTING BODY

The surface fitting method was also applied to part of the HL-10 lifting body. Figure 9 shows a view of this body and the six cross sections used to generate the surface fit. Tabulated values of the coordinates of surface points were available for many cross sections in addition to the six actually used, and these additional cross sections were used as a check on the accuracy of the surface fit. Due to symmetry about the  $\bar{x}, \bar{y}$  plane, only half of the body was used.

Figure 10 illustrates the first attempt to curve fit the cross-sectional data at  $\bar{x} = 128.32$ . Only two segments were used, and hence the resulting curve-fit is not satisfactory. Figure 11 shows the revised curve-fit to this same cross section when three segments are used, and the results are very good.

After satisfactory curve-fits were obtained for all six cross sections, using three segments in each one, the cross sections were blended in the longitudinal direction by use of the parametric cubic splines. Since each three-dimensional longitudinal curve is represented by its projections in the  $\bar{x}, \bar{y}$  and  $\bar{x}, \bar{z}$  planes, a total of  $2^4$  planar curves were used to represent the four longitudinal curves required for each of the three segments in a cross section. Of these  $2^4$  curves, seven of the initial spline-fits were found to have wiggles, and hence they were modified by specifying their slopes at the nose ( $\bar{x} = 0$ ). Some of the revised longitudinal curves are shown in figure 9, and they are actually smooth although the plotter makes them appear to have wiggles.

As a check on the accuracy of the final surface fit, cross sections were calculated at axial stations in between those used to generate the surface fit. Figure 12 compares the calculated cross section at  $\bar{x} = 43.656$  with the actual coordinates for this body. Other cross sections were found to compare even more closely than this one with the actual coordinates.

#### CONCLUDING REMARKS

An algorithm has been developed for surface fitting three-dimensional bodies from data points in several cross-sectional planes. This method was found to give satisfactory surface fits to the  $70^\circ$  slab delta wing and HL-10 lifting body. The surface geometry can best be analyzed through the use of an interactive computer graphics environment.

After a satisfactory surface fit to a body has been obtained, a relatively simple geometry subroutine package can be formed to use in other computer programs requiring a mathematical model of the geometry. It will calculate the body coordinates, slopes, and second derivatives at any position on the body. This method could be used to generate the input data for Coons' pathing method if desired. Computational time is short, and the amount of storage required is relatively small. A copy of the computer program can be obtained from the authors.

#### ACKNOWLEDGMENTS

This research was supported by NASA Langley Grant No. NGR 34-002-193. The NASA Technical Officer for this grant was Mr. H. Harris Hamilton of the Advanced Entry Analysis Branch, Space Systems Division.

## APPENDIX A

### Solution for Conic Section Equation

Substitute Eqs. (2), (3), (7), and (8) into Eq. (1) to obtain the conic section in the local coordinate system in the form

$$C_j z^2 + P_j z + Q_j = 0 \quad (A1)$$

where

$$P_j = - \left[ (A/m)_j + (A/n)_j \right] y + (A/m)_j y_j \quad (A2)$$

and

$$Q_j = A_j y \left( y - y_j \right) . \quad (A3)$$

The solution of Eq. (A1) is

$$\begin{aligned} z &= \left[ -P_j \pm R_j^{1/2} \right] / 2C_j \quad \text{for } C_j \neq 0 \\ z &= -Q_j / P_j \quad \text{for } C_j = 0, P_j \neq 0 \end{aligned} \quad (A4)$$

where the discriminate ( $R_j$ ) is given by

$$R_j = P_j^2 - 4C_j Q_j . \quad (A5)$$

In order to obtain real roots for  $z$  from Eq. (A4), Eq. (A5) must give  $R_j \geq 0$  for  $0 \leq y \leq y_j$ . Note that Eq. (A5) gives  $R_j \geq 0$  when  $A_j C_j \geq 0$ .

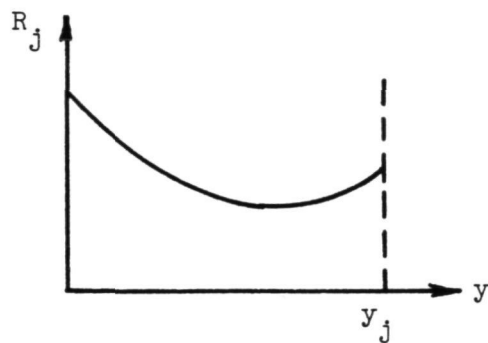
Also,

$$R_j = (A/m)_j^2 y_j^2 \geq 0 \quad \text{at } y = 0$$

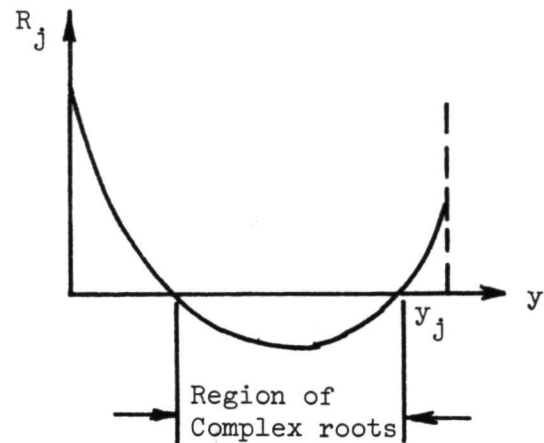
and

$$R_j = (A/n)_j^2 y_j^2 \geq 0 \quad \text{at } y = y_j .$$

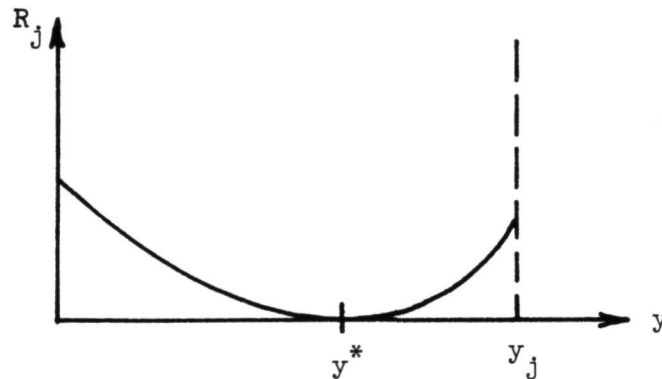
The possible variations for  $R_j$  are illustrated on the following page.



$R_j > 0$ , real roots



The minimum value of  $A_j C_j$  which still gives real roots corresponds to the limiting case shown below.



The minimum value of  $R_j$  occurs at  $y = y^*$  where both  $R_j = 0$  and  $dR_j/dy = 0$ . The solution of  $R_j = 0$  and  $dR_j/dy = 0$ , using Eq. (A5), gives the minimum value of  $A_j C_j$  as

$$\left(A_j C_j\right)^* = (A/m)_j (A/n)_j \quad (A6)$$

Thus, Eq. (A4) will give real roots for  $z$  if

$$A_j C_j \geq (A/m)_j (A/n)_j . \quad (A7)$$

The choice of the sign to be used in Eq. (A4) is determined by requiring the equation to be satisfied at the control points ( $z = 0, y = 0$ ) and ( $z = 0, y = y_j$ ). For the first control point,  $y = 0$ , Eq. (A4) yields

$$z = \frac{-(A/m)_j y_j \pm \left[(A/m)_j^2 y_j^2\right]^{1/2}}{2C_j} \quad (A8)$$

and thus  $z = 0$  requires the + sign if  $(A/m)_j > 0$  and the - sign if  $(A/m)_j < 0$ . At the second control point,  $y = y_j$ , Eq. (A4) yields

$$z = \frac{(A/n)_j y_j \pm \left[ (A/n)_j^2 y_j^2 \right]^{1/2}}{2C_j} \quad (A9)$$

and  $z = 0$  requires the - sign if  $(A/n)_j > 0$  and the + sign if  $(A/n)_j < 0$ . These conditions are all compatible because a conic section passing through the two control points and single-valued in  $z$  will have  $n_j < 0$  if  $m_j > 0$  and  $n_j > 0$  if  $m_j < 0$ . The sign given by these conditions can be used in Eq. (A4) for all values of  $y$  in the range  $0 \leq y \leq y_j$ .

## APPENDIX B

### Least-Squares Solution for Conic Section Coefficients

Equation (14) cannot be satisfied at all data points because there would be more equations than unknowns. It is also not desirable, in general, to have the curve go through all the data points because there may be scatter in the data. Accordingly, Eq. (14) is applied at data point  $k$  and rewritten as

$$\alpha_{j,k} A_{j-1} + \beta_{j,k} A_j + \gamma_{j,k} A_{j+1} + z_k^2 C_j = r_{j,k} \quad (B1)$$

where  $\alpha_{j,k}$ ,  $\beta_{j,k}$ , and  $\gamma_{j,k}$  are the values of  $\alpha_j$ ,  $\beta_j$ , and  $\gamma_j$  evaluated at  $y = y_k$ ,  $z = z_k$  in segment  $j$ ; and  $r_{j,k}$  is called the residual. The least-squares solution of the overdetermined system of equations determines the coefficients  $C_1$ ,  $C_j$ ,  $A_j$  ( $j = 2, \dots, N$ ) which minimize the sum of the residuals squared (See ref. 8). Define  $K_j$  as the data point number which corresponds to the first control point in segment  $j$ . Square Eq. (B1) and sum over all the data points in continuous segments  $j = 1, \dots, N$  to obtain

$$\sum_{j=1}^N \sum_{k=K_j}^{K_{j+1}} \left[ \alpha_{j,k} A_{j-1} + \beta_{j,k} A_j + \gamma_{j,k} A_{j+1} + z_k^2 C_j \right]^2 = \sum_{j=1}^N \sum_{k=K_j}^{K_{j+1}} r_{j,k}^2 \quad (B2)$$

The right side of Eq. (B2) is minimized by the system of equations obtained by setting partial derivatives of Eq. (B2) with respect to the independent coefficients equal to zero. The result of setting partial derivatives with respect to  $A_2, \dots, A_N$  equal to zero yields the following set of equations:

$$\begin{aligned} a_{m,1} A_{m-2} + a_{m,2} A_{m-1} + a_{m,3} A_m + a_{m,4} A_{m+1} + a_{m,5} A_{m+2} + a_{m,6} C_{m-1} + a_{m,7} C_m \\ + a_{m,8} C_{m+1} = 0 \quad m = 2, \dots, N \end{aligned} \quad (B3)$$

The result of setting partial derivatives with respect to  $C_1, \dots, C_N$  equal to zero gives the following additional system of equations:

$$b_{m,1}A_{m-1} + b_{m,2}A_m + b_{m,3}A_{m+1} + b_{m,4}C_m = 0$$

$$m = 1, \dots, N. \quad (B4)$$

The combined system of Eqs. (B3) and (B4) gives a system of  $(2N - 1)$  linear equations for  $(2N - 1)$  coefficients. The parameters used in Eqs. (B3) and (B4) are defined as follows:

$$a_{m,1} = \sum_{k=K_{m-1}}^{K_m} \alpha_{m-1,k} \gamma_{m-1,k} \quad (B5)$$

$$a_{m,2} = \sum_{k=K_{m-1}}^{K_m} (\beta \gamma)_{m-1,k} + \sum_{k=K_m}^{K_{m+1}} \alpha_{m,k} \beta_{m,k} \quad (B6)$$

$$a_{m,3} = \sum_{k=K_{m-1}}^{K_m} \gamma_{m-1,k}^2 + \sum_{k=K_m}^{K_{m+1}} \beta_{m,k}^2 + \sum_{k=K_{m+1}}^{K_{m+2}} \alpha_{m+1,k}^2 \quad (B7)$$

$$a_{m,4} = a_{m+1,2} \quad (B8)$$

$$a_{m,5} = a_{m+2,1} \quad (B9)$$

$$a_{m,6} = \sum_{k=K_{m-1}}^{K_m} z_k^2 \gamma_{m-1,k} \quad (B10)$$

$$a_{m,7} = \sum_{k=K_m}^{K_{m+1}} z_k^2 \beta_{m,k} \quad (B11)$$

$$a_{m,8} = \sum_{k=K_{m+1}}^{K_{m+2}} z_k^2 \alpha_{m+1,k} \quad (B12)$$

$$b_{m,1} = a_{m-1,8} \quad (B13)$$

$$b_{m,2} = a_{m,7} \quad (B14)$$

$$b_{m,3} = a_{m+1,6} \quad (B15)$$

$$b_{m,4} = \sum_{k=K_m}^{K_{m+1}} z_k^4 \quad (B16)$$

Note that if slopes should be specified at selected control points, then the least-squares solution described above is applied to the segments between two consecutive control points with specified slopes. In the solution of the combined system of Eqs. (B3) and (B4) the term  $\alpha_{N+1,k}$  and  $\gamma_{N+1,k}$  must be interpreted as zero. Hence, the coefficients  $A_0$ ,  $A_{N+1}$ , and  $A_{N+2}$  do not appear in

the resulting system, and recall that  $A_1 = 1$  unless the conic section requires  $A_1 = 0$ .

#### REFERENCES

1. DeJarnette, F. R.: Calculation of Inviscid Surface Streamlines and Heat Transfer on Shuttle Type Configurations. NASA CR-111921, Aug., 1971.
2. Rakich, J. V., and Kutler, P.: Comparison of Characteristics and Shock Capturing Methods with Application to the Space Shuttle Vehicle. AIAA Paper No. 72-191, Jan., 1972.
3. Bartlett, D. A.: Computer Utilization for Aircraft Contour Determination. Society of Automotive Engineers, 700202, National Business Meeting, Wichita, Kansas, March 18-20, 1970.
4. Coons, S. A.: Surfaces for Computer-Aided Design of Space Forms. AD663504 MAC-TR-41, MIT, June, 1967.
5. Bezier, P.: Numerical Control Mathematics and Applications. John Wiley and Sons, 1972.
6. Craidon, C. B.: A Computer Program for Fitting Smooth Surfaces to an Aircraft Configuration and Other Three-Dimensional Geometries. NASA TM X-3206, June, 1975.
7. Mason, T. E., and Hazard, C. T.: Brief Analytic Geometry. Ginn and Co., 1947.
8. Scheid, F.: Theory and Problems of Numerical Analysis. Schaum's Outline Series, McGraw-Hill Book Co., 1968.
9. Ahlberg, J. H., Nilson, E. N., and Walsh, J. L.: The Theory of Splines and Their Applications. Academic Press, 1967.

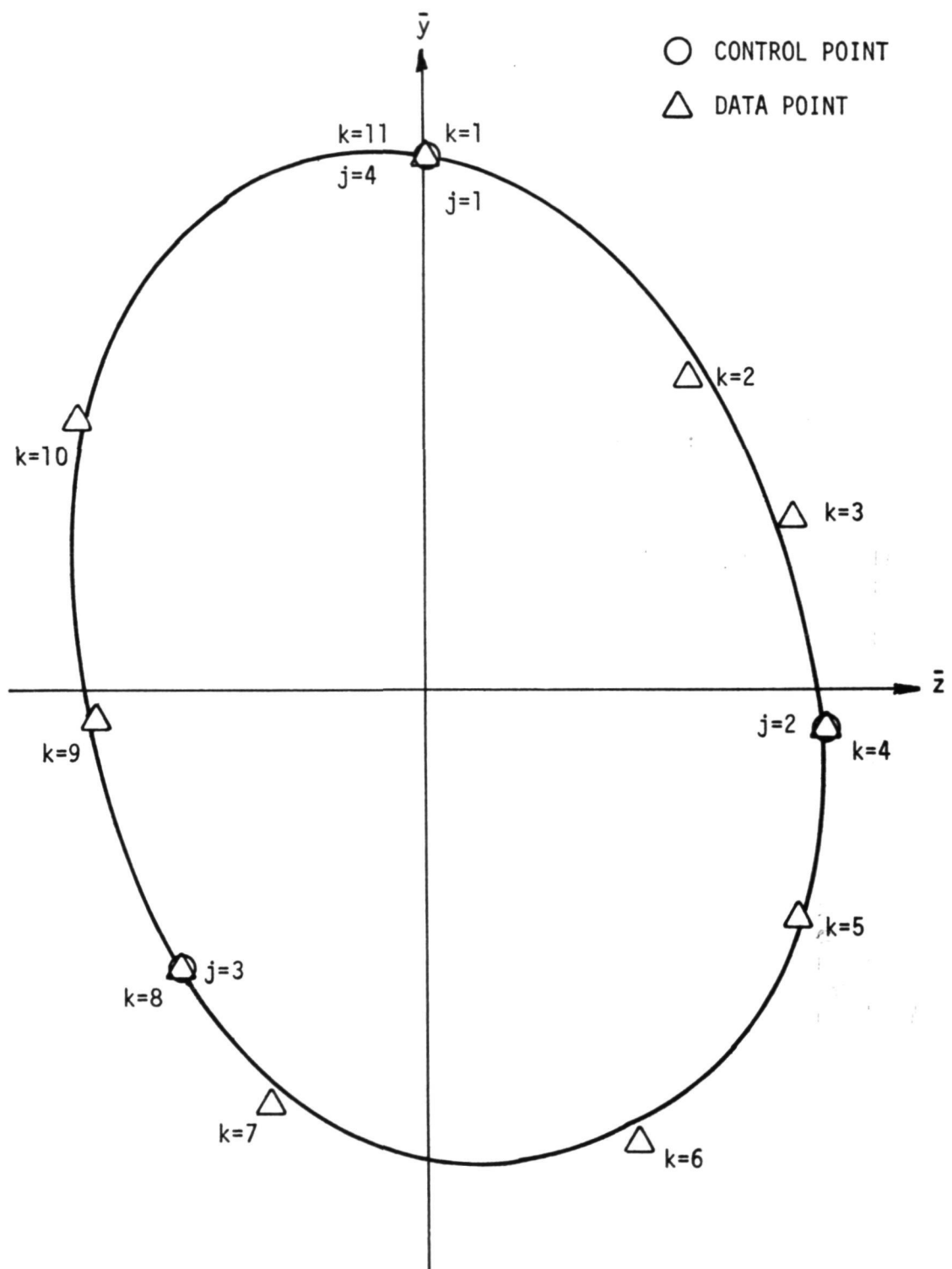


Figure 1. Control points and data points in a cross section.



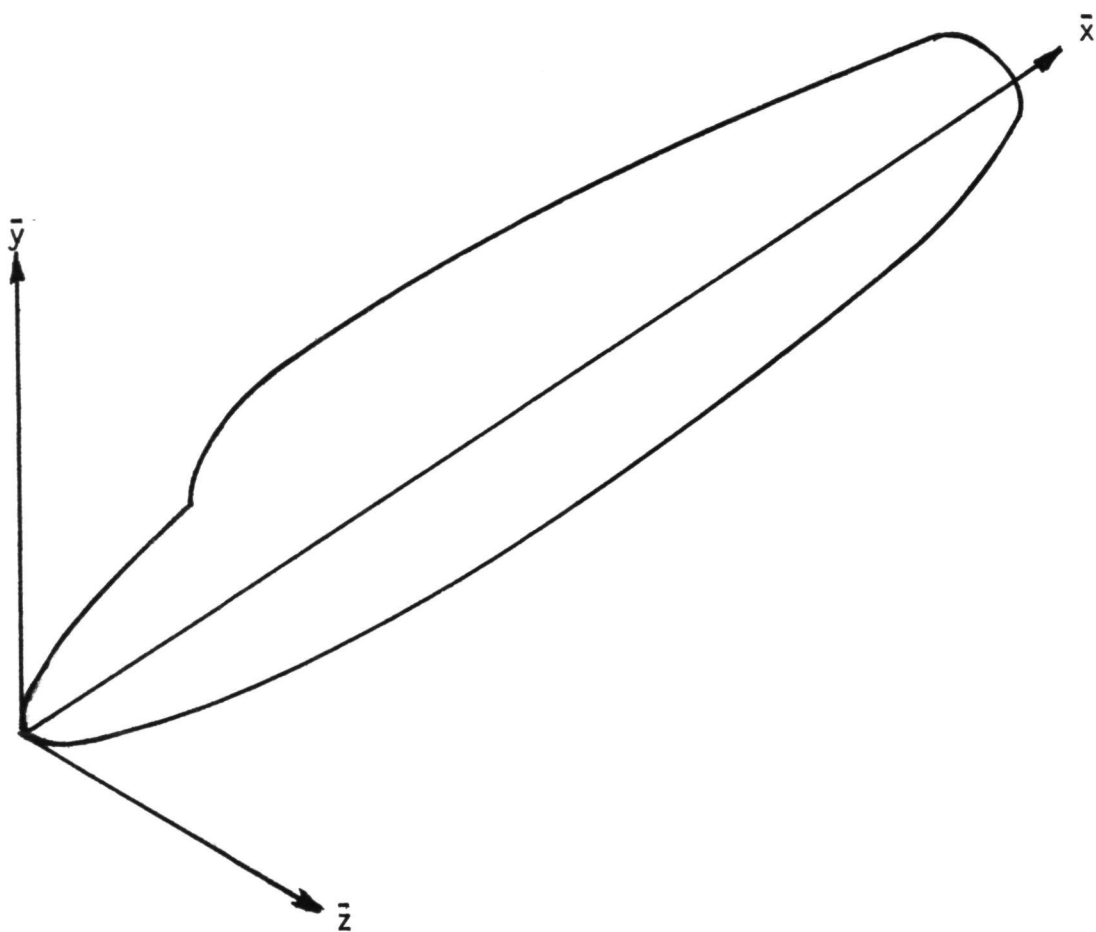


Figure 2. Cartesian coordinate system.

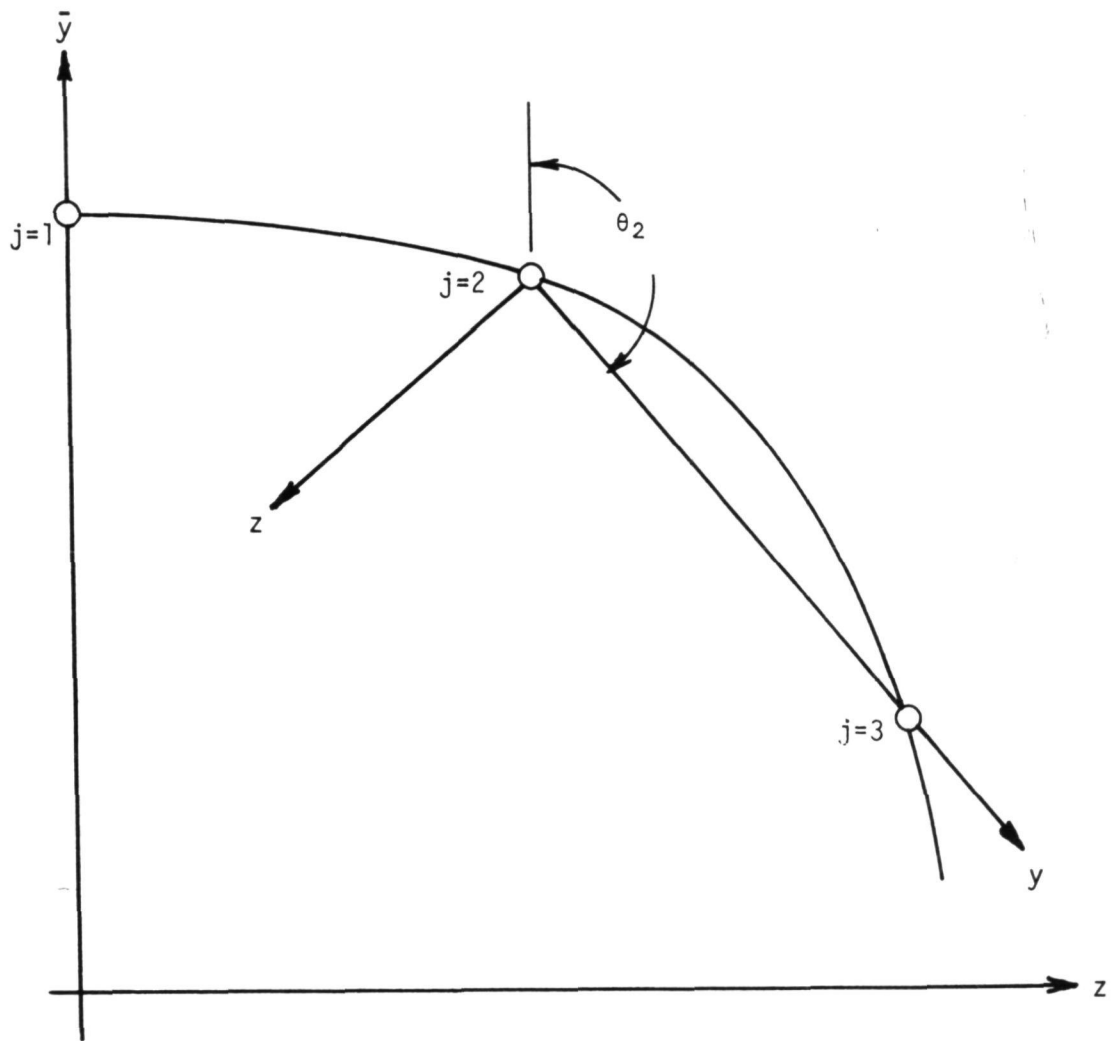


Figure 3. Local coordinate system, illustrated for segment  $j=2$ .

# Equation for General Conic Section

$$A_j y^2 + B_j yz + C_j z^2 + D_j y + E_j z + F_j = 0$$

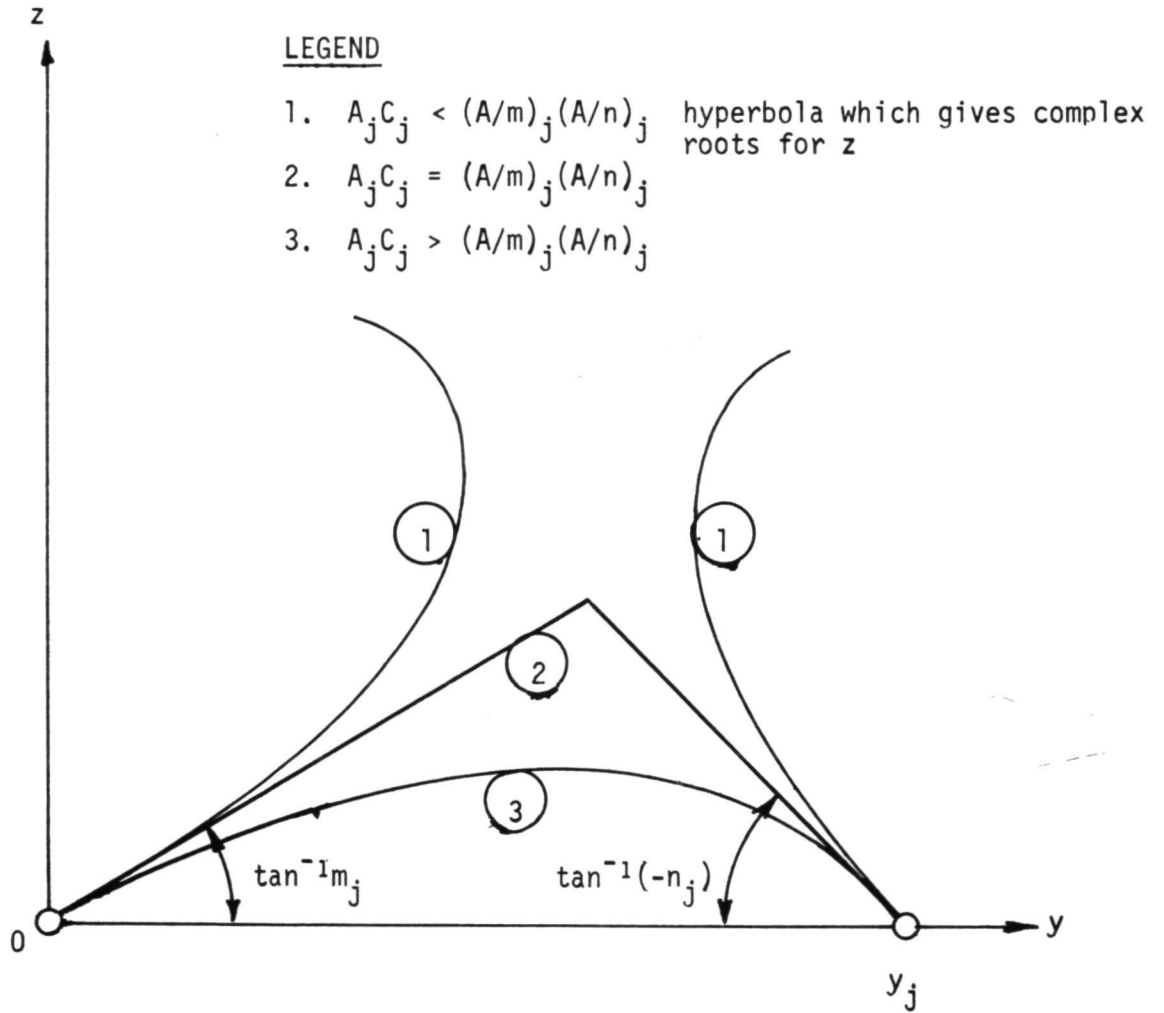


Figure 4. Effect of  $A_j C_j$  on conic section.

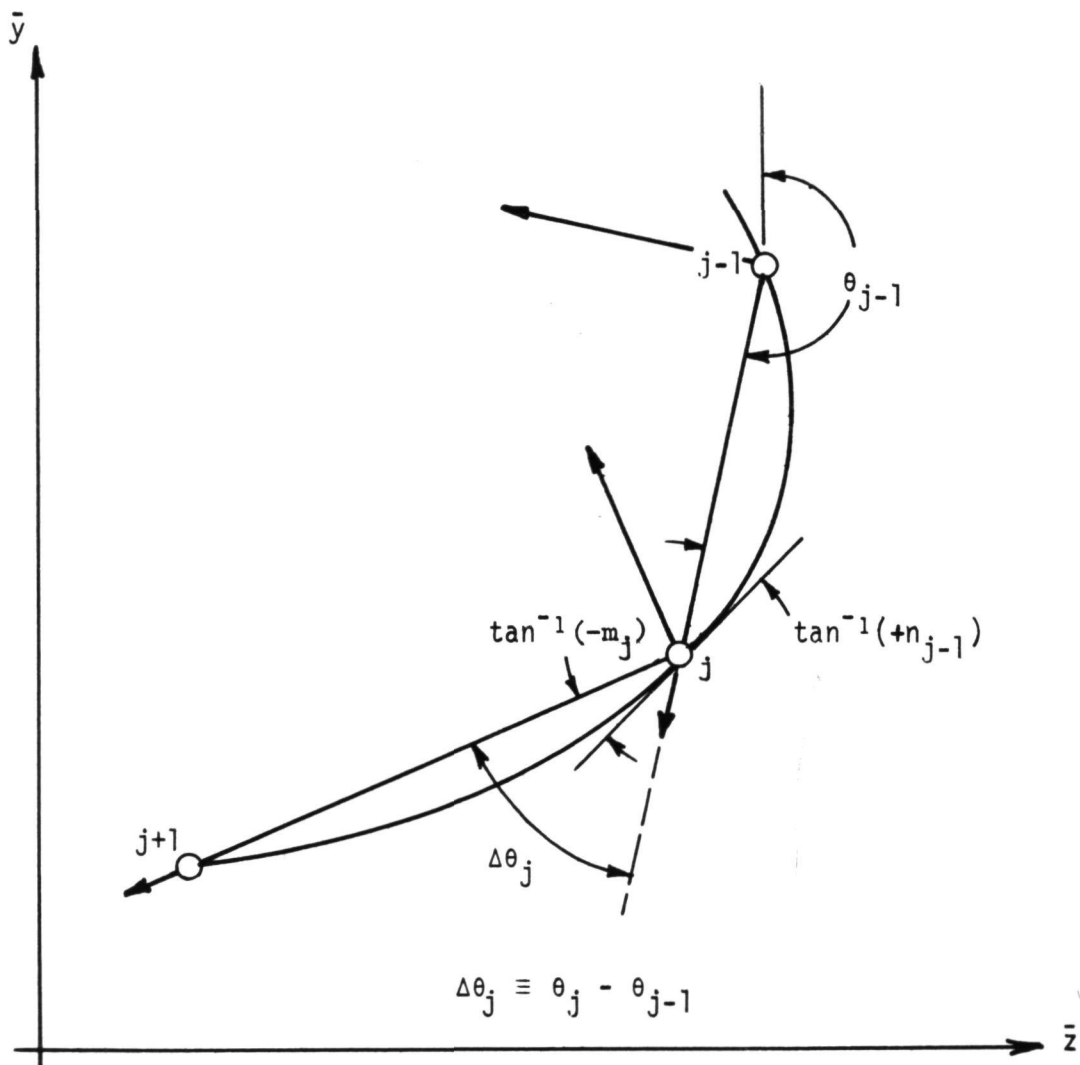


Figure 5. Continuity of slope at a control point.

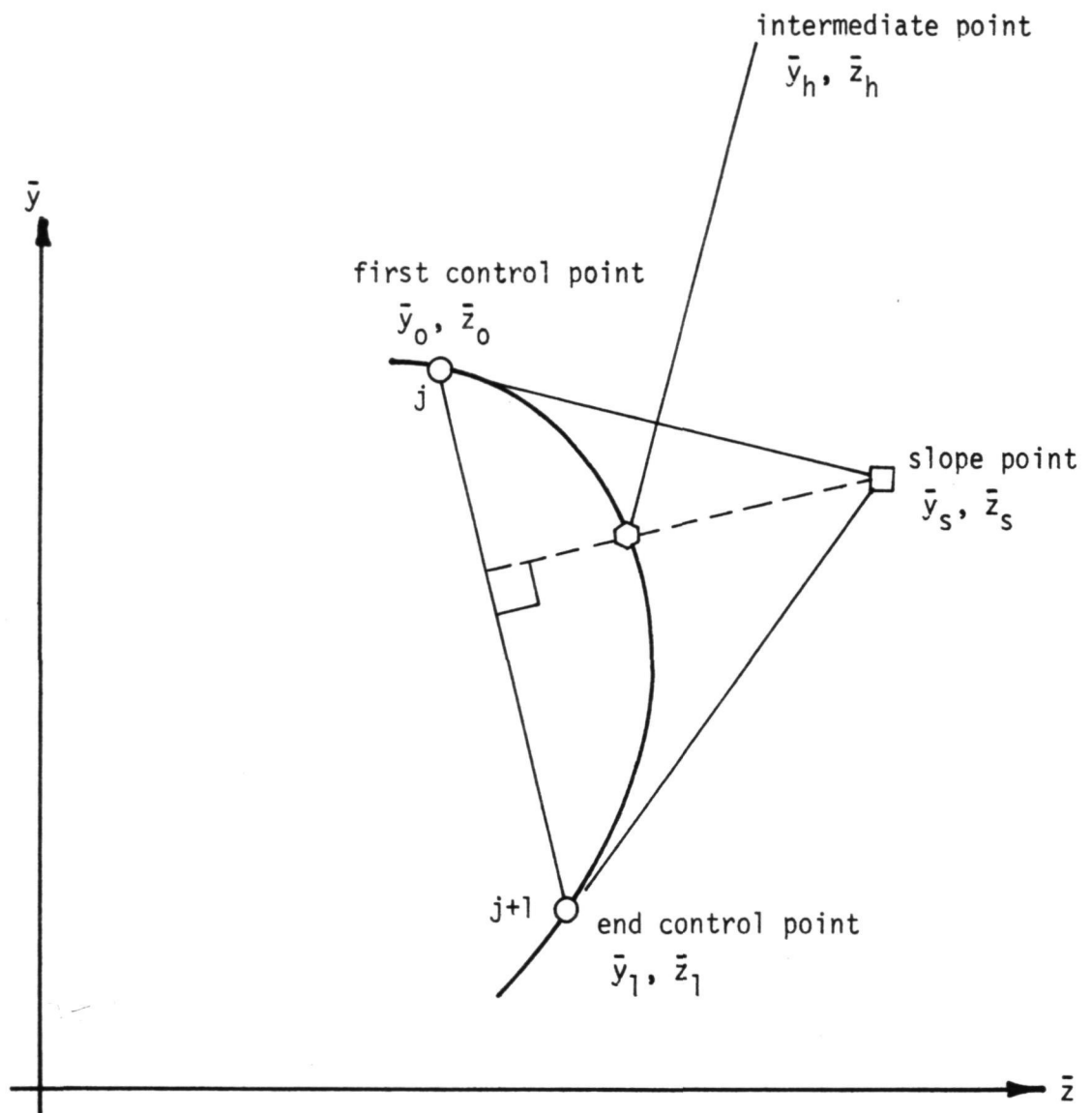


Figure 6. Four points used to define a segment of a conic section.

- CONTROL POINT,  $(\bar{y}_0, \bar{z}_0)$  and  $(\bar{y}_1, \bar{z}_1)$
- ◻ INTERMEDIATE POINT,  $(\bar{y}_h, \bar{z}_h)$
- SLOPE POINT,  $(\bar{y}_s, \bar{z}_s)$

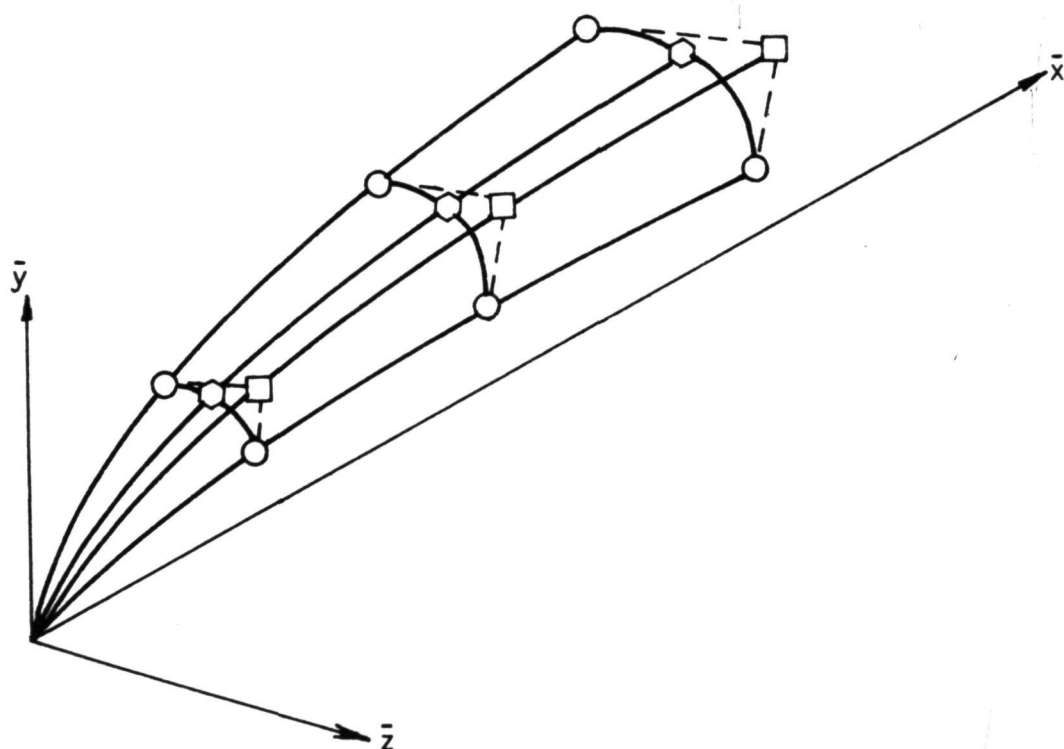
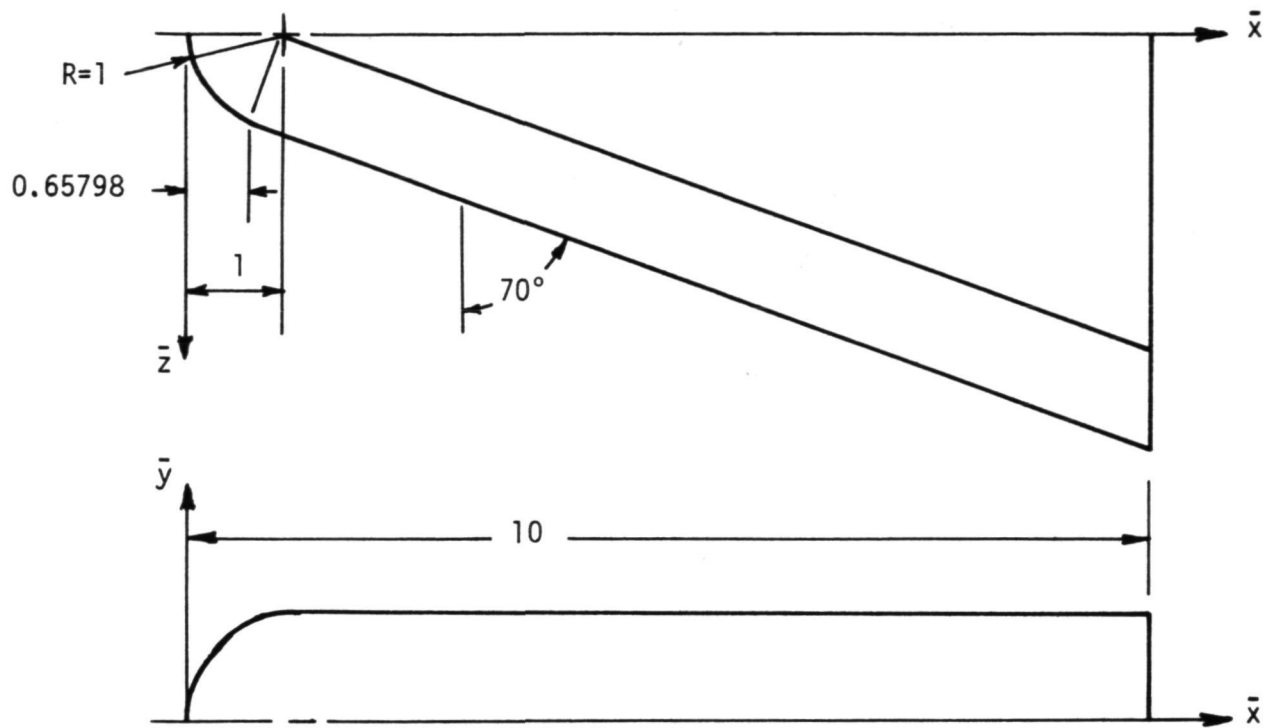


Figure 7. Longitudinal curves through the four points used to define a segment of a conic section.



○ CONTROL POINT

△ DATA POINT

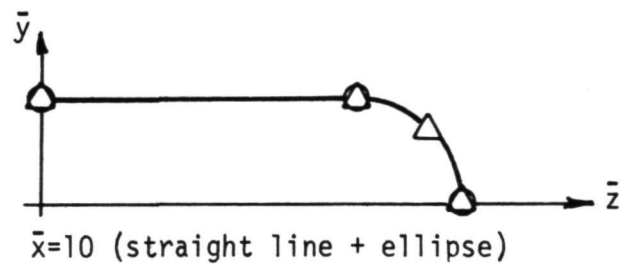
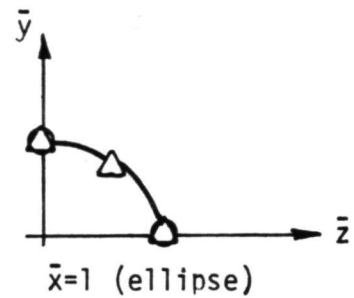
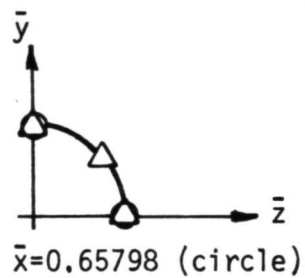


Figure 8. Geometry of 70° delta wing.

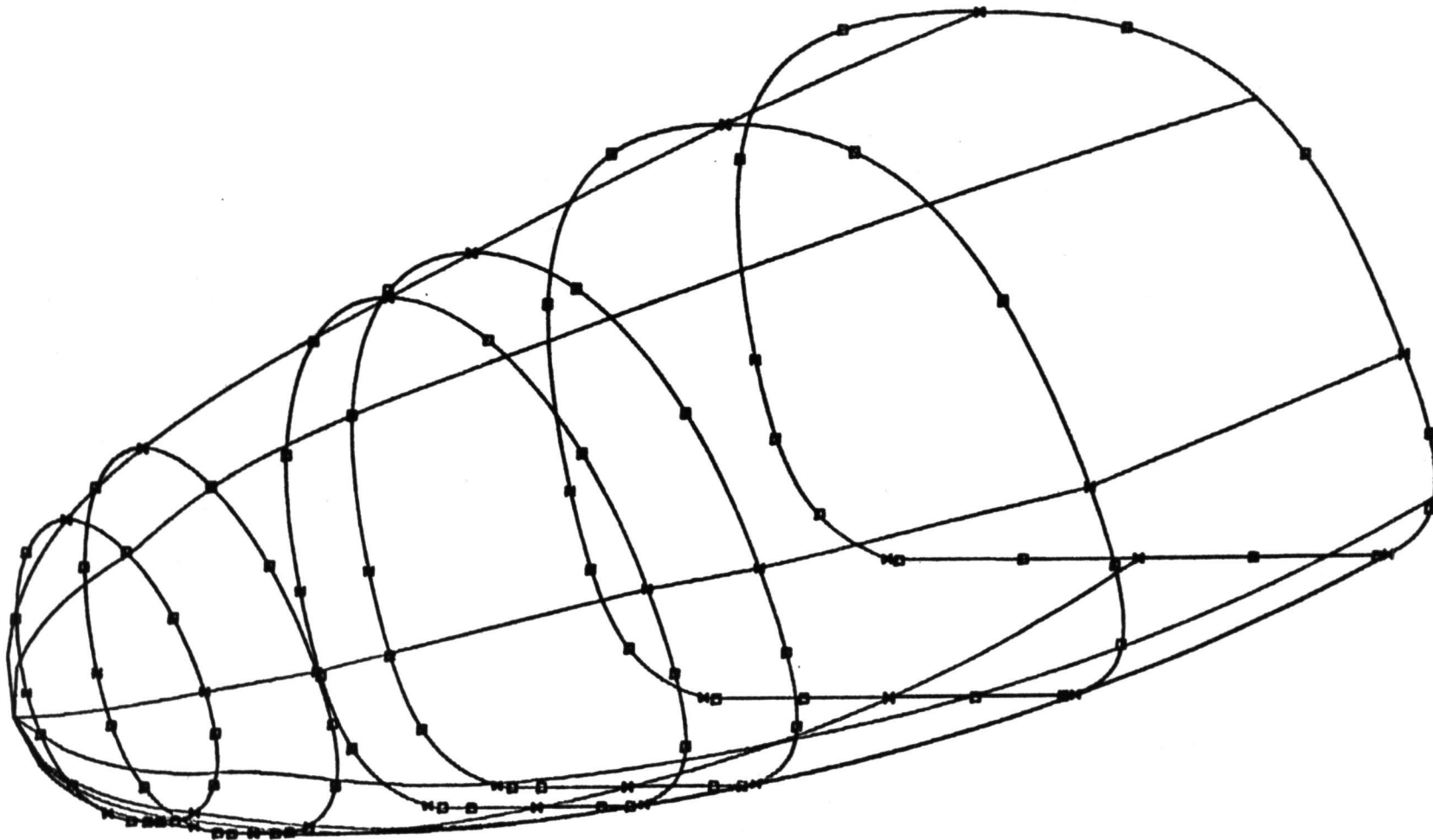


Figure 9. View of cross sections and longitudinal curves for HL-10 body.



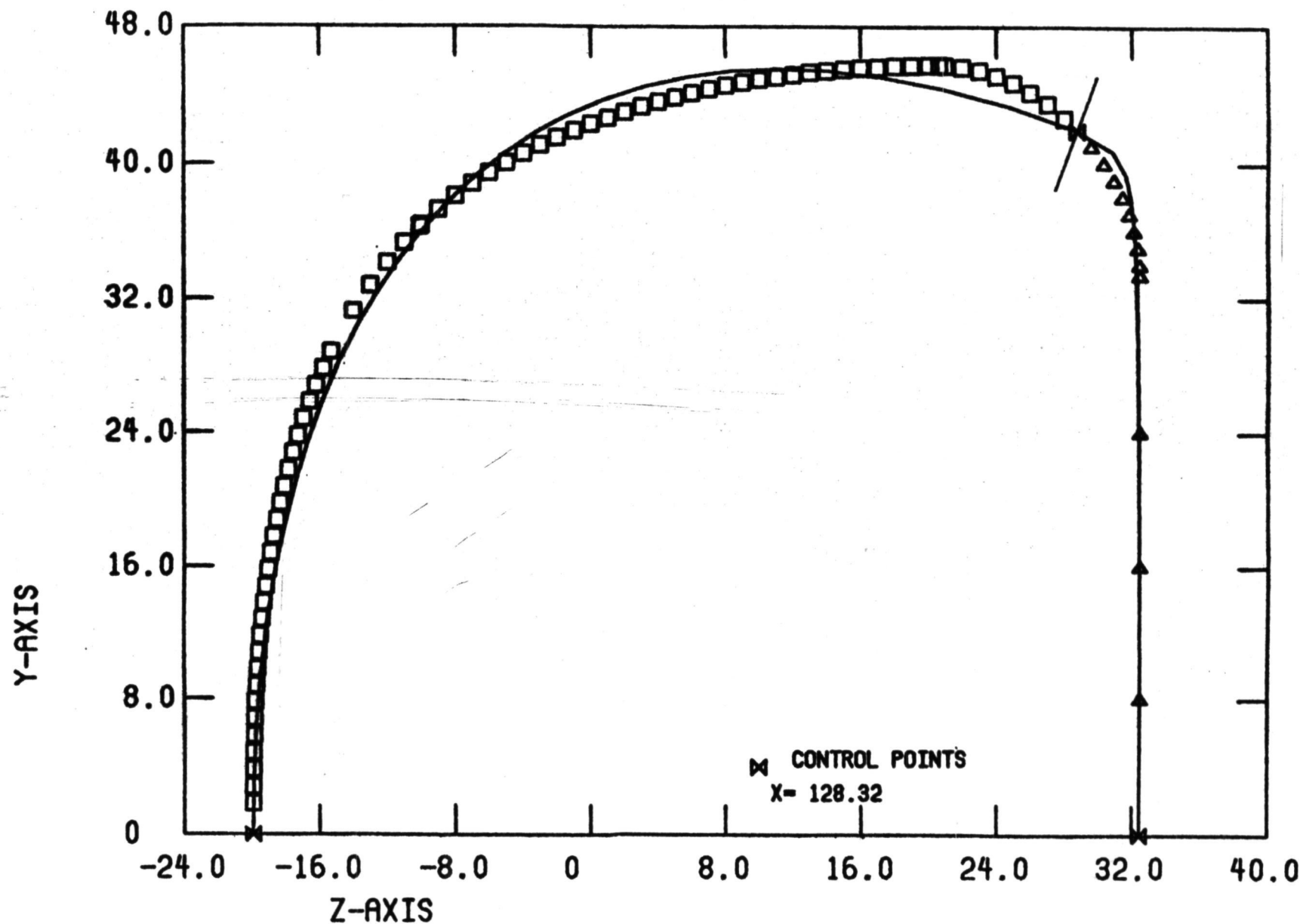


Figure 10. First attempt to curve-fit cross section of HL-10 body using two segments.

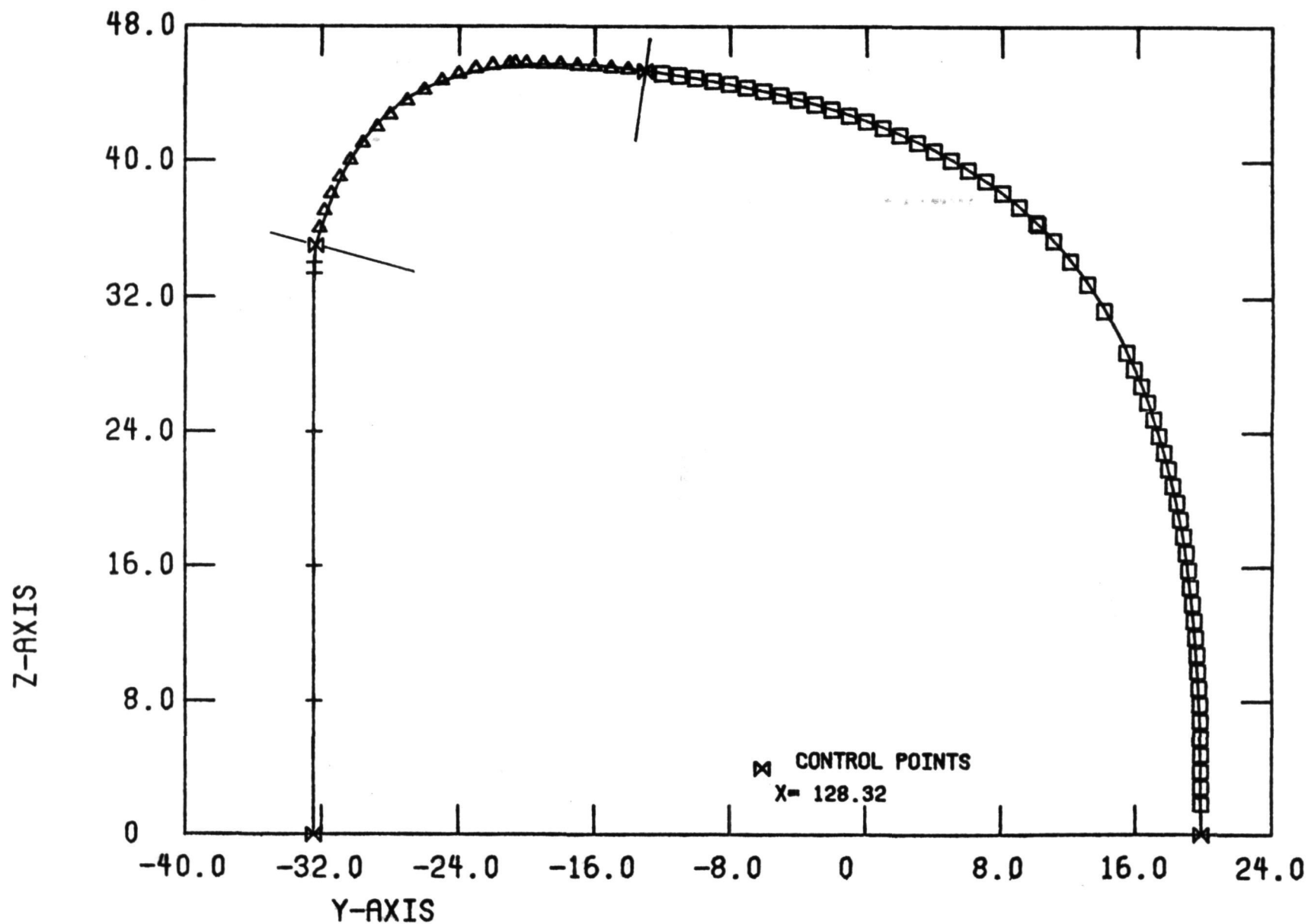


Figure 11. Revised curve-fit of cross section of HL-10 body using three segments.

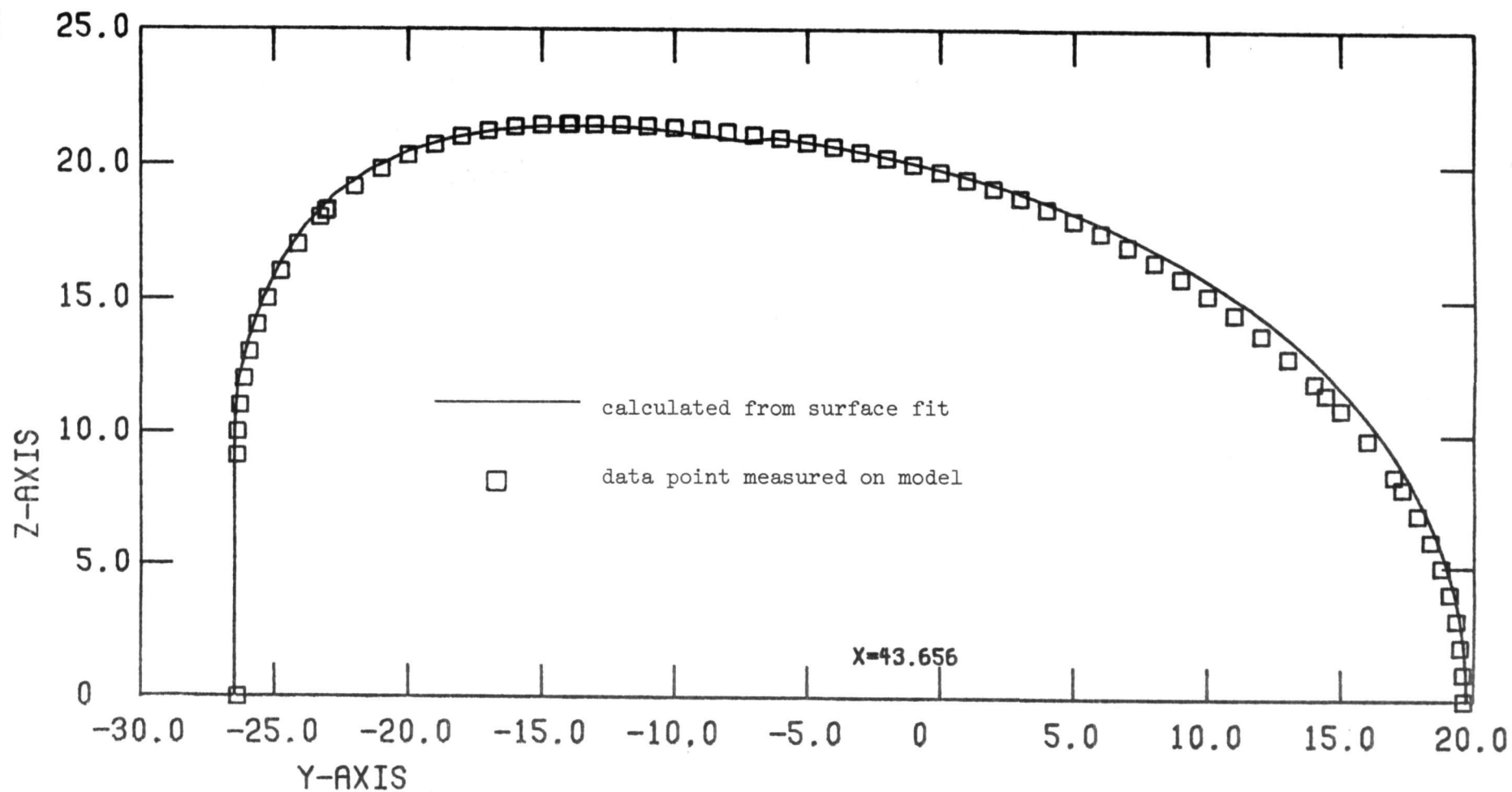


Figure 12. Comparison of cross section calculated in between input cross sections with measured data on HL-10 body.

ENGINEERING COMPUTER GRAPHICS IN GAS TURBINE ENGINE DESIGN,  
ANALYSIS AND MANUFACTURE

Richard S. Lopatka

Pratt & Whitney Aircraft

ABSTRACT

This paper serves as an overview of a time-sharing and computer graphics facility designed to provide effective interactive tools to a large number of engineering users with varied requirements. The evolution of applied interactive graphics at Pratt & Whitney Aircraft will be traced as it occurred within the engineering environment. The application of computer graphics displays at several levels of hardware complexity and capability will be discussed; with examples of graphics systems tracing gas turbine product development, beginning with preliminary design through manufacture. Highlights of an operating system stylized for interactive engineering graphics will be described. P&WA problems and solutions in supplying cost effective, timely, and easily usable graphic tools to a wide spectrum of engineering users provide insight for others supplying or about to apply interactive graphics in similar environments.

BACKGROUND

Prototype interactive graphics work began at P&WA in the mid 60's on a UNIVAC 1556 display located at the United Aircraft Research Laboratories. Using an AED based, high level graphics language and a roll in/roll out modification to the operating system, several practical applications of interactive graphics were demonstrated to management. These applications could be categorized as enhancing analytical tasks in the design process by graphical visualization of input/output, by editing of key input parameters and by monitoring and modifying iterative processes to optimize solutions. The potential of interactive graphics and computing was evident by these first attempts despite operational, hardware, and organizational handicaps. A decision was made in 1968 to install an IBM 360/67 with three 2250-3 displays and 15 conventional typewriter terminals for time sharing. Application system and graphics program development was begun 6 months prior to delivery on a 360/50 supporting a 2250-1 display.

Impetus for the first integrated design system at P&WA came from a requirement to reduce design and analysis flow time in turbine blade and vane durability analysis. The Turbine Airfoil Development System, TADSYS, was implemented under CP/OS and introduced key concepts of dynamic module selection and execution from a CRT plus an online data base facilitating communication between application modules within the system (figure 1). After one year, results from using TADSYS proved that engineering analysis time could be reduced by a factor of 6, and that the decision-making capabilities of the engineer were expanded to achieve better designs through more detailed analyses. The success of TADSYS soon prompted development of a similar system for compressor aerodynamics (COMP) completed in mid 1970 and achieving similar results.

The introduction of interactive graphics provided many challenges to the computing team. A transitional period was necessary when compatibility with

batch methods was maintained. A modular programming approach allowed portions of the integrated system to be used early in development. Inexperience in graphics and human factors engineering were evident until the system support matured and graphic programs showed the logic stability and display quality necessary in interactive graphic applications.

As computer graphics was developing at P&WA, the computing support organization was also taking form. Computer graphics programming was begun, and experience developed, in a small task group, but is now a standard programming tool for the entire application programming section. A factor affecting the impact of computer graphics at P&WA is a single, complete computing organization working closely together, and including operations, systems analysis, machine systems, and all application programming. This group of approximately 300 people has a sole objective to service the needs of P&WA Engineering. Its success is measured in terms of its impact on engineering projects. Programming, done by "professionals" in the application programming group, has grown in complexity not only in computer graphics but in data management and systems integration. Programming is recognized as a discipline requiring a full-time trained specialist in order to best use state-of-the-art computer hardware and software.

Projects are established and funded on an individual basis. The need for, and basic approach to, a task is determined by a user-programmer team. Development is a joint effort with an end-user representative playing a major part in task specifications, including graphic frame content and final program verification. This environment appears particularly conducive to development of interactive graphics, which requires not only proficiency in computer programming, but also a clear understanding of human factors and detailed user needs.

#### COMPUTER GRAPHICS FACILITY

Main support for engineering at P&WA consists of two IBM S/370-168s. One computer is devoted entirely to time-sharing including computer graphics during normal working hours and is operated under VM/CMS. The other machine operates under OS/VS2 and supports scientific batch and engineering commercial applications. Communication between the two 168s is an integral part of the interactive facility.

As stated previously, productive use of interactive graphics started with IBM 2250 displays. Widespread implementation of graphics at P&WA continued with the introduction of low-cost storage tube displays. Total analysis and design systems previously developed in their entirety for refreshed displays were re-analyzed, decomposed, and graphics modules implemented where most effective. Many passive or limited interactive graphics modules are run as effectively on storage tube displays and at a 10 to 1 cost differential in terminal costs. Storage tube displays are located directly in engineering work areas and provide, as needed, hands-on flexibility -- not possible with the limited number of more expensive devices requiring days-ahead scheduling for individual use. For example, a general object breakup program, GOB, has been implemented at both the refreshed display and storage tube level. Initial finite-element modeling for structural and thermodynamic analyses is completed at a

2250 display using the full picture update and light pen interaction capabilities of a refreshed display. Component trade studies requiring shape modifications, boundary condition application, banding optimization, and the overall first model verification are completed. Generally, follow-on engineering iterations necessary during component analysis require less geometric viewing and modification, and are processed on the storage tube using the same online files generated from the 2250 display.

This philosophy of a hierarchy of terminal capabilities has been extended to many areas of engineering application including preliminary design specification and engine simulations (SOAPP), component and engine test data reduction (MDR), and component design and analysis (DERVA, VIBRA). Presently, in support of these application systems, there are over 80 addressable ports, available under VM/CMS. Most are alphanumeric CRTs or edit typewriter terminals but over 30 graphic devices are supported. At present 24 Tektronix 4010s, 8 Tektronix 4014s, 3 IBM 2250s and one E&S Picture System (manufactured by the Evans and Sutherland Computer Corporation) complete the terminal complement. Offline plotting facilities including Calcomp drum plotters, and a Gerber flatbed model. Current plans include installation of a Computervision 3D-IDS system for generating detail drawings and a minicomputer-based engine part hardware measuring inspection device for online digitizing and comparisons with nominal design coordinates.

An important aspect of a computer graphics facility is software support -- particularly at the systems and graphics processing levels. Considerable effort at P&WA is directed toward making the time-sharing and graphic systems usable by noncomputer experts and engineers, without the assistance of programmers or computing technicians. Much of this software uses VM/CMS as a base. CP/CMS, its forerunner on the 360/67, was developed by IBM initially as an aid in operating system development. However, the command language flexibility and simplicity, virtual machine structure and time-sharing supervisor provide a ready foundation for an engineering-oriented time-sharing network (reference 1). The total system can be viewed as having five major software operating levels -- system supervisor, virtual machines, subsystem interfaces, exec procedures, and application modules (figure 2).

#### System Supervisor

At the systems supervisor level, graphics performance has been improved by modifications to the paging algorithm, shared pages, locking high activity pages, and by a special interrupt handler geared to P&WA's mix of compute bound and interactive requirements. This tuning has been critical to the success of time-sharing at P&WA because totally integrated engineering tasks, composed of interaction and CPU processing, are typically completed during a terminal session.

#### Virtual Machines

The virtual machine structure under VM affords the graphics user functions which enhance the usability of the terminal hardware. The virtual machine principle, stated simply, is one in which each user of the system has access to all components of a real computer; i.e., console terminal, memory, disc, and unit

record devices. The Control Program (CP) manages all requests and usage of real devices in the system. From the view of the terminal user he is using a (virtual) computer devoted entirely to his task. He is, in one respect, the computer operator. The command set allows him to start and stop tasks, attach discs, and access I/O devices at will. The P&WA conversion of the Graphics Access Method (GAM) to operate under CMS has given the 2250 user all the advantages of CMS. Prior to 1972, GSP/GAM was supported by an OS/PCP virtual machine running under CP and the engineer user was obliged to act as an OS machine operator. The user of a virtual machine is able to access and update data files, invoke language processors (APT, FORTRAN, COBOL, SCRIPT, SNOBOL) and run complete applications directly in his pre-assigned virtual machine.

Virtual machines also provide additional support functions which affect user productivity. Once a virtual machine is logged-on, the terminal console device may be disconnected, leaving the virtual machine operational but unattended. In this mode, certain communication functions are handled and special output is produced. Hard copy plotting mainly on offline Calcomps is generated by transferring output plot data directly from one user virtual machine to the virtual reader of a special disconnected plot virtual machine. Machine operations personnel transfer (dump) this plot data to magnetic tape when spool activity dictates. By this means, hard copy plot documentation is returned, in a timely fashion, to the terminal user with little inconvenience or effort on his part. Similar procedures exist for providing punch tapes for numerical control machine tools and computer output on microfiche (COM).

Disconnect virtual machines also serve a performance monitor function and are an integral part of a Central Data Base facility, to be covered later in this paper.

### Subsystems

Virtual machines serve as communicators for subsystems (figure 3). Remote job entry (RJE) and cross machine communications are handled with disconnected virtual machines. Extensions to the CMS command set provide establishment of OS job streams and allow for submissions to the batch S/370-168 during a terminal session. The full command set of a HASP RJE station is available from any virtual machine and is regularly used to query status of individual jobs within the batch computer.

Approximately 300 jobs/day are being processed via this link during normal periods. Communication with the PDP-11 based Picture System is currently supported at the virtual RJE level; a higher level FORTRAN-callable (SEND/RECV) protocol has been defined, and will be implemented shortly. This facility will provide much needed software support, whereby the CPU power and file management capability of the 168 can be combined with the picture quality and dynamics of a high performance graphic display, and exercised concurrently, during a Picture System terminal session. In general, off-the-shelf minicomputer operating systems do not address themselves to man-machine human factors encountered in a CAD/CAM application. Communications with S/370-168 need to be completed before the full potential at P&WA for this high performance display is realized.

## Exec Procedures

Supported within VM/CMS is a (exec) procedure generation capability, which allows basic CP and CMS commands to be grouped together and thereby generate a macro level command (reference 2).. This exec facility contains looping commands, argument lists, and stack procedures which are extremely useful in creating end-user task procedures; which make transparent to end-users many of the basic machine system commands required for computer communication. Exec writing has become an important level of application programming in P&WA design systems. They are used to control program selection and execution in integrated design systems; they establish I/O protocols with system resources, as well as with user online files throughout the system. Designers and engineers, for the most part, communicate with the computer via these execs and see few of the basic CMS command set.

## Program Modules

Application modules are written almost exclusively in FORTRAN and complete the system. A library of approximately 500 distinct modules is operational on the time-sharing system. These programs are used throughout the design process and in many areas of manufacturing. Figure 4 depicts the areas using the interactive facility and illustrates the availability of a shared data base for interchange of engineering design data from discipline to discipline. This integration of the entire computerized design process is an underlying objective of systems development and is proceeding in an evolutionary manner. The basic computer concepts involved will be presented later. P&WA's experience has been primarily in design areas. Figure 5 shows those portions of gas turbine engine design which utilize interactive graphics for design analysis support within an integrated design system.

In many cases, the application graphics codes include use of many special purpose routines which have been developed to supplement vendor supplied software. In particular, routines to perform some functions in the non-intelligent 2250 display processor have reduced the number of interrupts to the mainframe. Faster response in light pen tracking, scaling, and some real-time object translations have enhanced graphic interaction.

## TURBINE AIRFOIL DEVELOPMENT SYSTEM

An example of an integrated design system, which serves to illustrate more concretely the significance of the total system environment, is the Turbine Airfoil Development System, TADSYS. A description with emphasis on engineering content and application can be found in a paper by K. Thomas and J. Piendel (reference 3). A more encompassing description of the turbine CAD/CAM system was presented by E. Nilson at the USA-Japan Design Automation Symposium '75 (reference 4). The major objective of the system, as it exists today, is to support rapid design and analysis of the turbine component in a gas turbine engine. Figure 6 shows the major steps in the process. Many of them contain several computer programs which are described at some length in reference 3.



Stepping through the process helps to show how computer graphics plays a role in this system.

The system begins with a one dimensional gas dynamic analysis of the entire turbine followed by a streamline determination and gas loading analysis. This is accomplished at the storage tube level. Graphic output of the flowpath and velocity vectors is possible (figure 7). The storage tube was selected because the input to these modules is mainly alphanumeric. However, passive graphical output is valuable in the iterative process of establishing a flow-path. Data resulting from these studies is deposited in a data base for input to individual airfoil aerodynamic design. This data together with parametric models of airfoil profiles is used to compute proposed airfoil contours (figure 8). The contour cannot be established until studies of the subsonic and transonic pressure distributions and boundary-layer effects are completed. Again the device is the storage tube display. A significant computation requirement exists at this step along with interactive graphics design needs. This dual requirement of heavy CPU load and interactive response is a difficult one to meet. In TADSYS, a background virtual machine processes computations while the designer is preparing new cases or examining previous results. This is one of the few areas in P&WA's design process that has needed this unique procedure.

After completion of the aerodynamic design, the terminal used for graphics becomes the 2250 refreshed display. An internal cooling scheme is synthesized at the display using the external shape as a starting point. A flow topology, representing all the internal flow paths, is determined and becomes the model for a detailed flow balance of the 3D flow network (figure 9). Conceptually, this has been one of the most difficult graphics processors we have had to develop. The graphics processor we use today has the ability to represent several path types including film holes, channels, orifices, pedestals and expansion-contractions. The designer interacts with the design shape on the CRT and the program computes automatically all characteristics of the flow path from the stored geometric model. Results of this analysis are stored as boundary conditions for subsequent steps in the process.

A finite-element model is needed for the heat transfer and stress studies which follow in the process. A family of interactive breakup programs, designed for turbine airfoil shapes, is available. The principle in all of them is identical. Given the outline of the cross section to be analyzed, the designer interactively selects flag points on the surfaces which determine major areas in the model where the coarseness of the model will be specified (figure 10). The grid (number of rows and columns) in each of these regions is then indicated. The modeling program computes the fine grid model and all properties required for thermal and stress analyses. The result is displayed on the CRT followed by key properties of the model (figure 11).

Both steady-state and transient thermal analyses are possible within the system. The designer supplies additional thermal inputs at the display and the computation begins. Output is displayed in isotherm form (figure 12) with interrogation methods provided interactively. Slices through the airfoil can be requested and temperatures at the neighboring nodes are displayed. Results of the following stress analyses use isoplot methods for stresses and strains

(figure 13). Scaling options and the ability to compute additional isolines on the fly enhance the effectiveness of these packages.

TADSYS is designed for simultaneous multi-terminal operation. Other graphic modules and many nongraphic interactive steps are part of the system. The advancing engineering technology of turbine design now requires frequent use of full 3D aerodynamic, thermodynamic, and structural analyses. Several current designs have utilized upgraded analyses and the data base has been used to extract these models.

#### CENTRAL DATA BASE

Systems like TADSYS have the ability to communicate with a Central Data Base, which is geared to assist in engineering data flow from discipline to discipline in the design and manufacturing process. This file management system relies on the virtual machine facility to handle data both in a preliminary design mode and in a final design sense (figure 14). A considerable amount of the data is geometric and represents component designs synthesized and analyzed at interactive graphic terminals. Part definition can migrate from preliminary design to manufacturing where process planners, tool designers, and numerical control part programmers use information of value to them.

The Central Data Base currently supports design systems at the file management level only. Data bases (as in TADSYS) address data at the record and word level. When data is transferred to Central, header and control information at the beginning of every file is stored in directories. Directories can be integrated by the user via search/sort utilities. Files are managed by an application system data base administrator whose responsibilities are to maintain data integrity and to monitor the proper balance of online and tape archival data.

The need for a technical data management system is crucial to any large-scale integration of interactive computing systems. VM/CMS - PWA Graphics does not totally meet this need. Systems to handle very large engineering/manufacturing data bases are under study. Preliminary work suggests that it will be possible to expand the present system, possibly using a Virtual Data Access Manager (VDAM) approach (reference 5) together with a direct communication support between active virtual machines.

#### HIGH PERFORMANCE GRAPHICS

The spectrum of computer graphics at P&WA has been upgraded to encompass a high performance intelligent display, namely the E&S Picture System. A requirement exists in design and manufacture for high resolution, real-time dynamics of 3D objects and distributed processing to support fast response in user interaction. Applications include pre- and post-processing of 3D structural, thermal and aerodynamic data, representation of hardware test rig results, geometry

synthesis of complex turbine blades and vanes and NC data generation and verification. Potential for advanced technology graphics at P&WA is highest in areas where costly and time consuming model making and trial part manufacture can be eliminated by computer models.

Figure 15 is a photograph from the CRT display showing the core of a turbine blade and an inserted tube partially removed. A simulation of the removal process is required to assure that the component can be manufactured. Twist and curvature of the part combined with tight tolerances in the tube operating position make this verification non-trivial. The high performance display's hardware clipping has been used as a plane slicer and linear interpolator such that displays of any section of the object in various orientations can be generated. The data structure within the computer is such that the pierce points can be reconnected and redisplayed in real time. This dynamic slicing has been valuable in this application as well as others. Figure 16 is a photograph from the CRT of a numerical control cutter path of approximately 5000 lines. This display has proved useful in determining errors in NC part generation, particularly for complex surfaces. The Picture System display program communicates with the main APT support under CMS. Storage tube graphics is also utilized in NC applications. Acknowledgement is given to L. C. Knapp from E&S for his onsite software development and system support in this Picture System effort.

#### SUMMARY

This paper presents an overview of a total system environment conducive to engineering use of computer graphics. The growth in computer graphics as illustrated in Figure 17 is some indication of the success of this effort. Records show that over 600 engineers and technicians per month use the VM/CMS - PWA Graphics facility -- nearly 900 have received formal training. What has evolved at P&WA is a result of many major contributions by people with special skills and talents working in an atmosphere which has made the user transitions to new computer technology and computer graphics viable.

## REFERENCES

1. McGrath, M., "Virtual Machine Computing in an Engineering Environment", IBM Systems Journal 11, No. 2, 131-149 (1972).
2. IBM Virtual Machine Facility/370: Exec User's Guide, GC20-1812-0, First Edition August 1973.
3. Thomas, K. M., Piendel, J. J., "An Automated Interactive Design System for Advanced Gas Turbines", Gas Turbine Conference and Product Show, Zurich, April 1974. ASME Preprint 74-GT-82.
4. Nilson, E. N., "The Pratt & Whitney Aircraft Interactive Turbine CAD/CAM System", USA-Japan Design Automation Symposium '75, Toyko, Japan, August 1975.
5. Warn, D. R., "VDAM - A Virtual Data Access Manager for Computer Aided Design", Workshop on Data Bases for Interactive Design, U. of Waterloo, Ontario, September 1975.

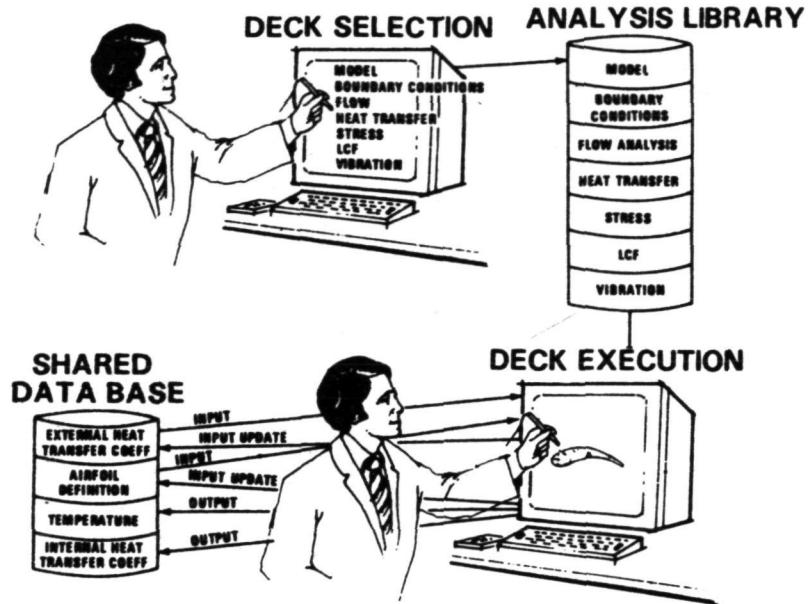


Figure 1.- Interactive design system operation.

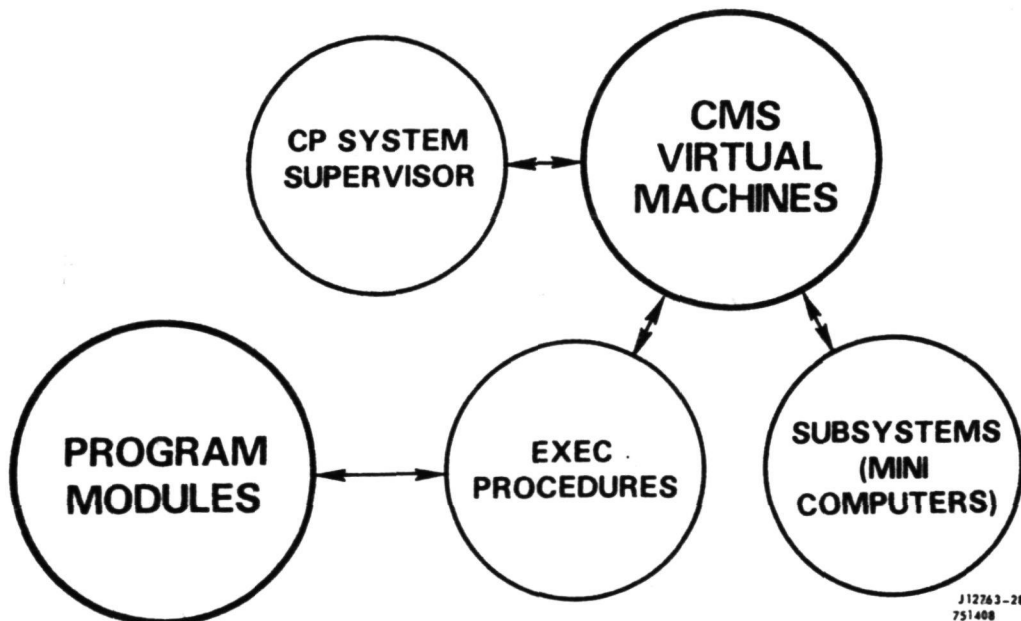


Figure 2.- VM/CMS P&WA graphics overview.

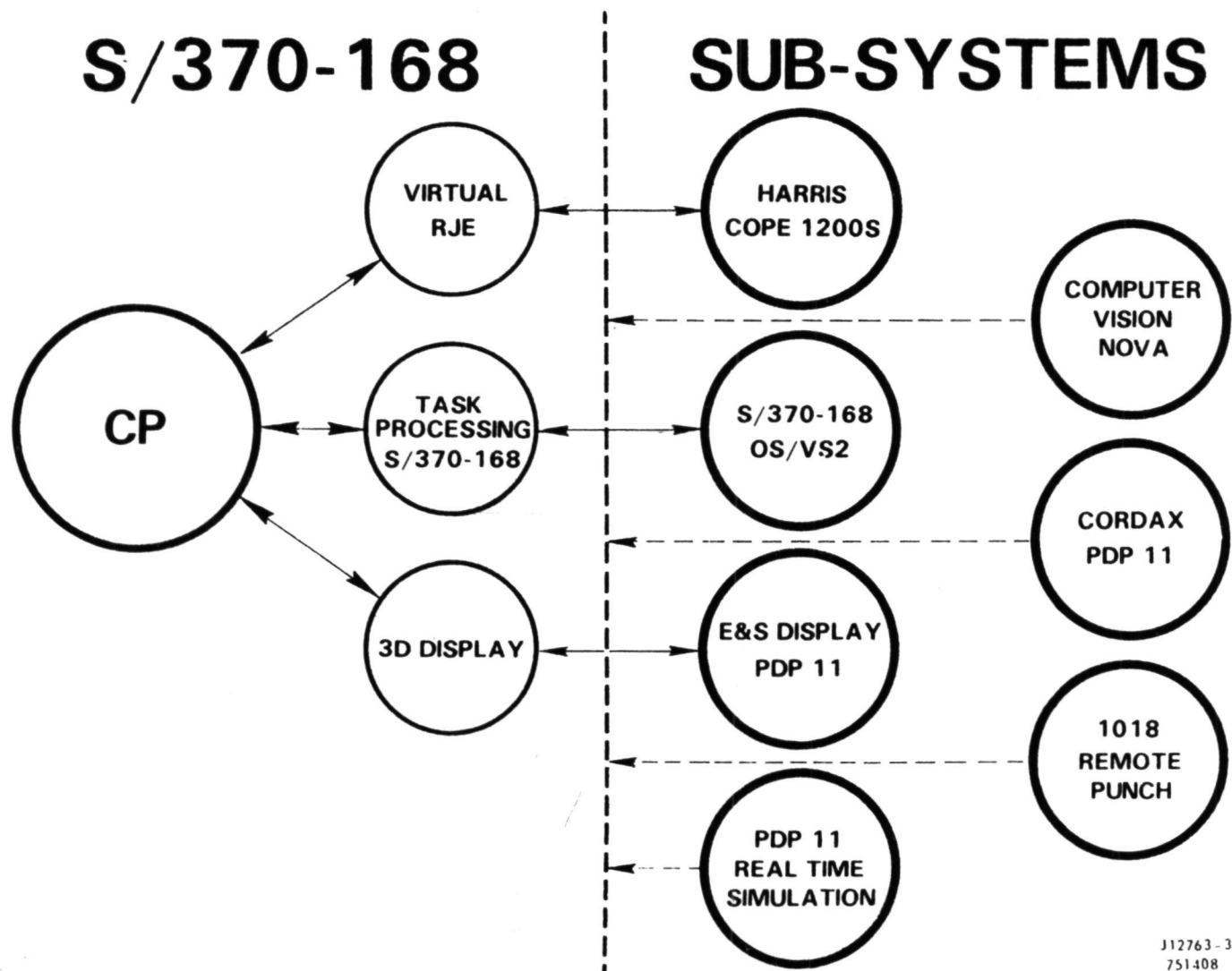
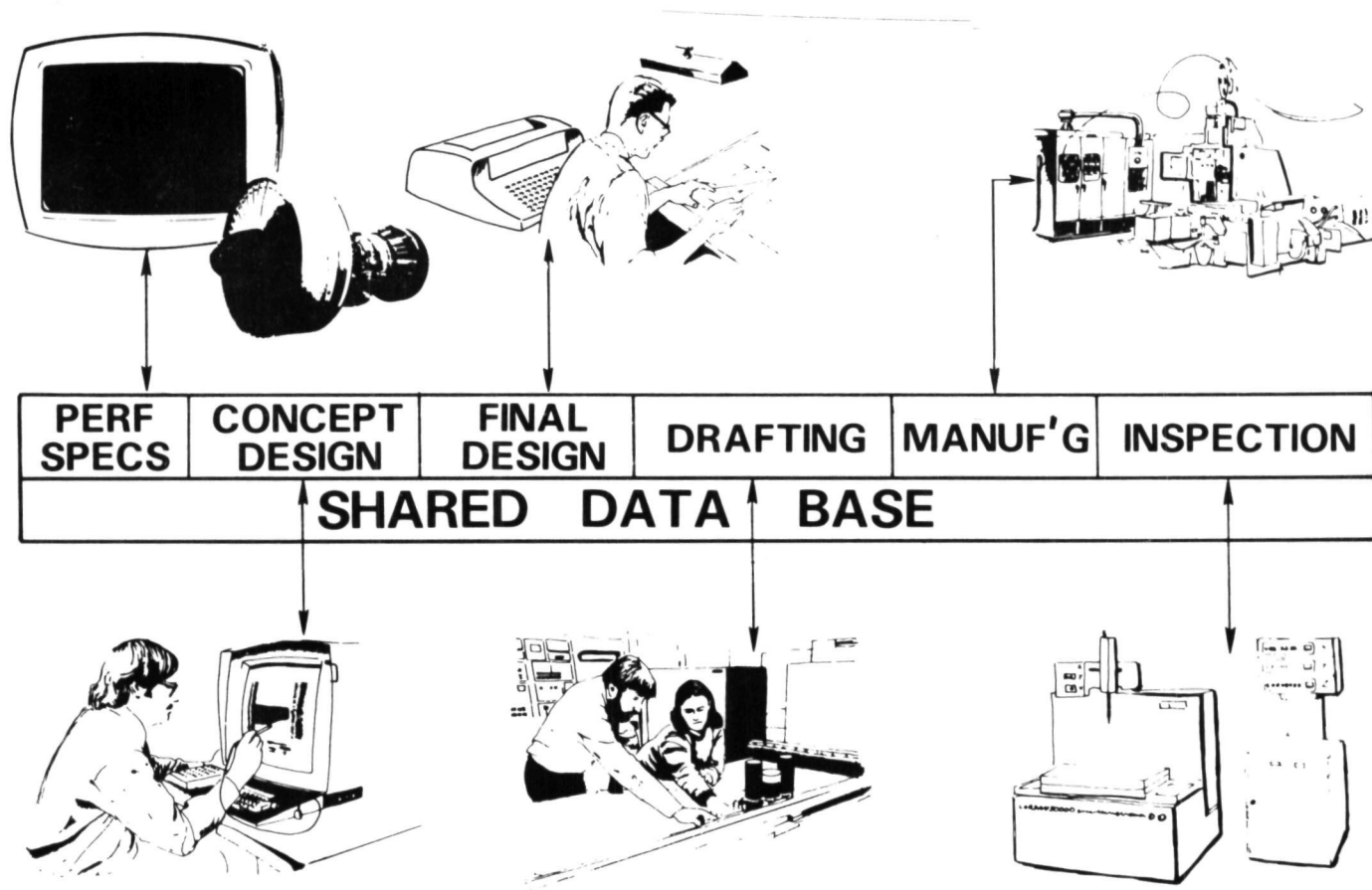


Figure 3.- Subsystems.

J12763-30  
751408



J12763-23  
751308

Figure 4.- CAD/CAM shared data base.

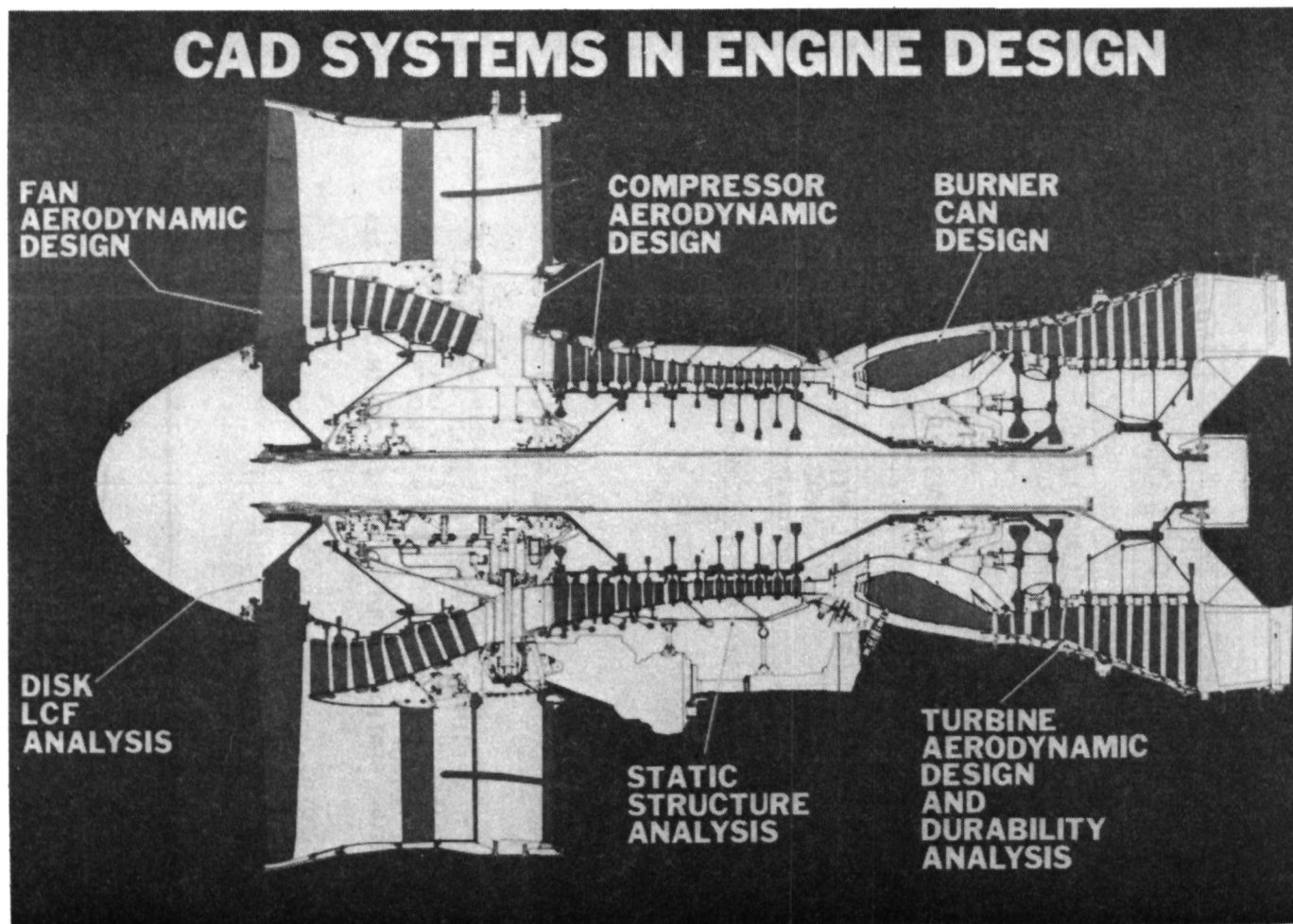
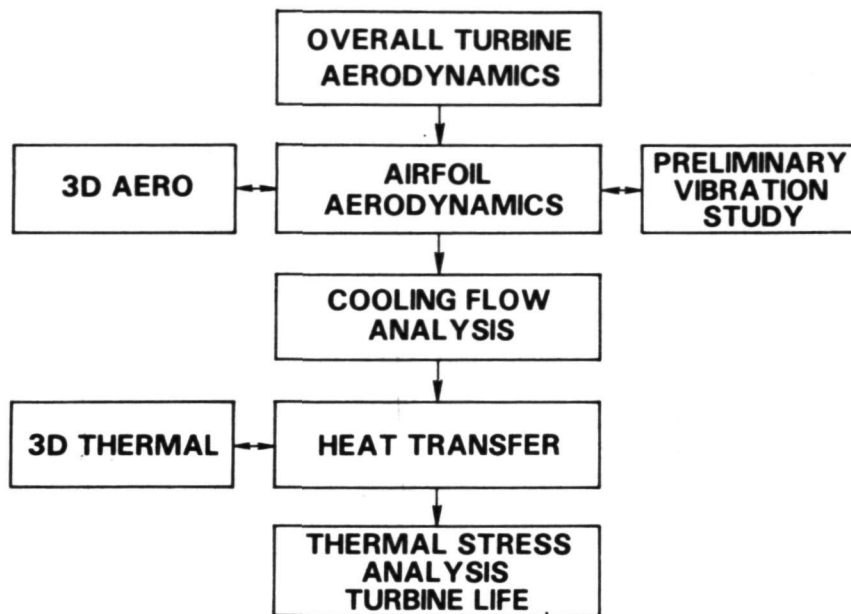


Figure 5.- Computer-aided design systems.





J12763-10  
751308

Figure 6.- Turbine airfoil development system.

W677 STREAMLINE ANALYSIS  
CONVERGED FLOWPATH  
OPTIONS:

- |   |                          |
|---|--------------------------|
| 3 BLOW UP USING CROSSHAIRS                      | 6 PRINTED OUTPUT OF CASE |
| 1 RETURN TO ORIGINAL FLOWPATH<br>(NOT BLOWN UP) | 7 DELTA OPTION           |
| 2 RETURN TO OUTPUT CONTROL                      | 8 RETURN TO INPUT        |
| 3 OVERALL OUTPUT                                | 9 EXECUTE CASE           |
| 5 INDIVIDUAL STATION OUTPUT                     |                          |
- \*\*\* TYPE IN CHOICE \*\*\*

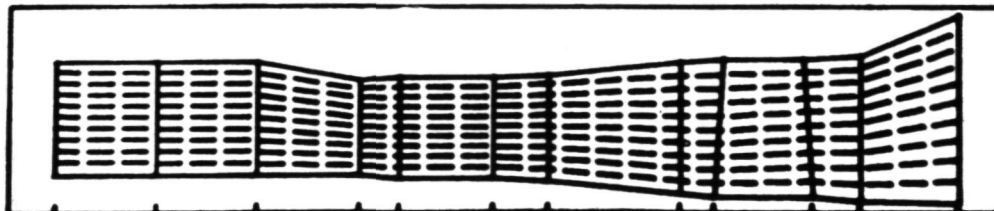


Figure 7.- Overall aerodynamics. Storage tube.

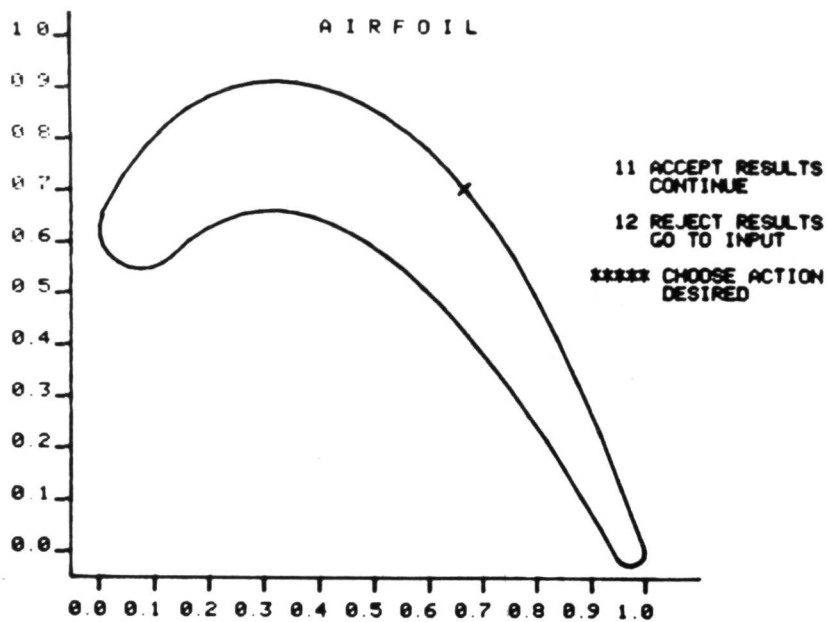


Figure 8.- Airfoil profile. Storage tube.

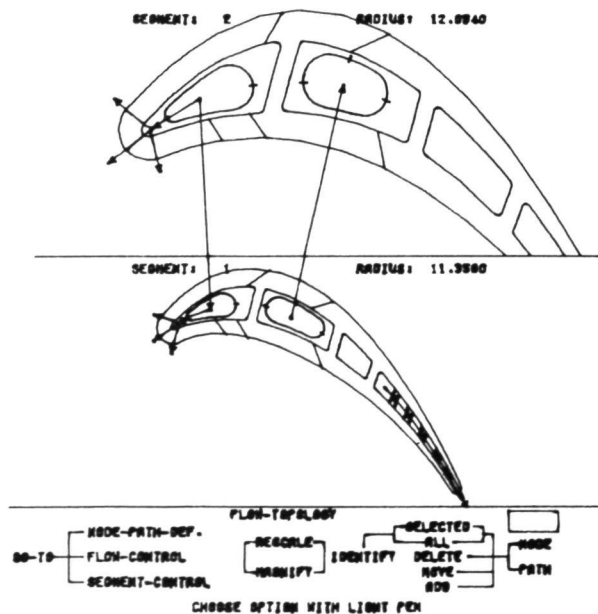
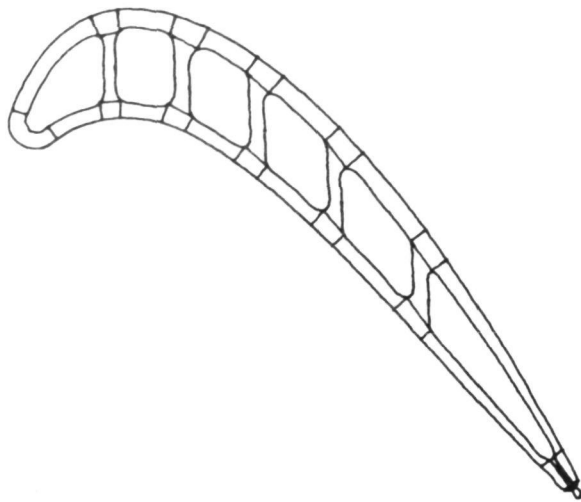


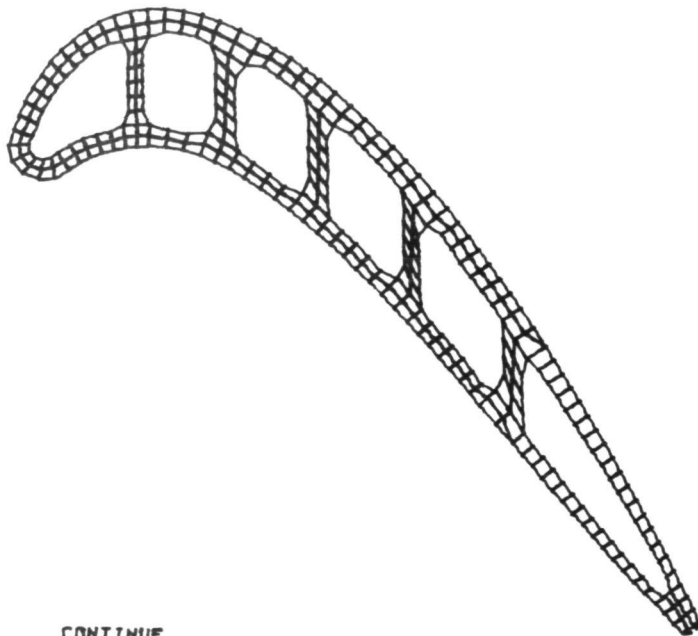
Figure 9.- Internal flow topology. Refreshed display.



SELECT NEW ROW AND COLUMN BREAK-UP

USE ROW AND COLUMN BREAK-UP IN LIBRARY

Figure 10.- Flag point description. Refreshed display.



CONTINUE

Figure 11.- Finite-element model. Refreshed display.

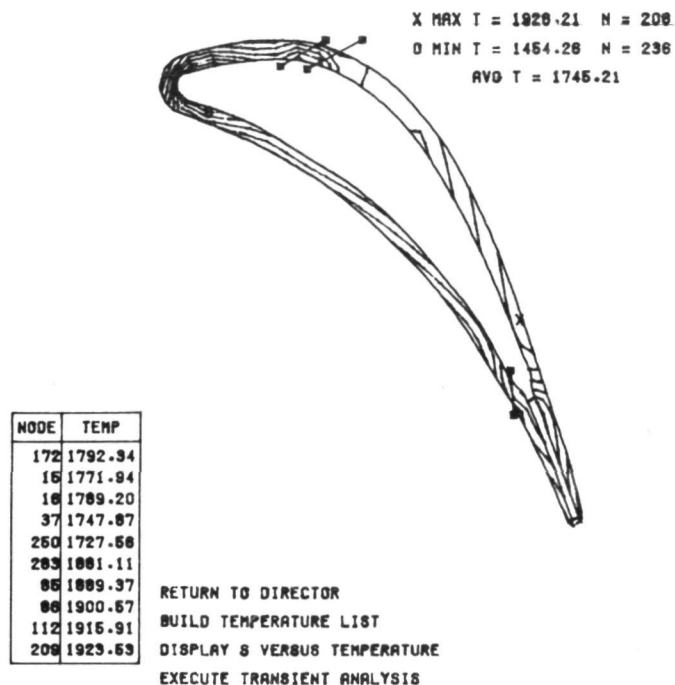


Figure 12.- Isotherm display. Refreshed display.

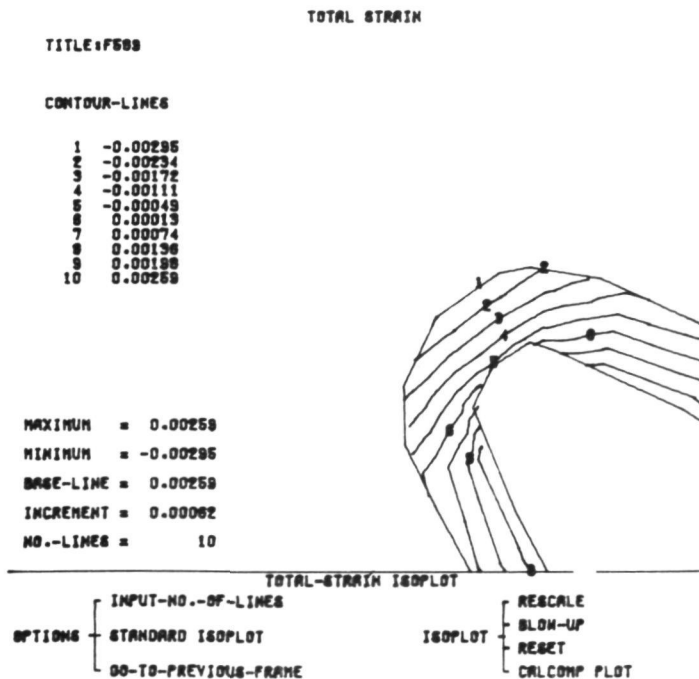


Figure 13.- Isostrain display. Refreshed display.

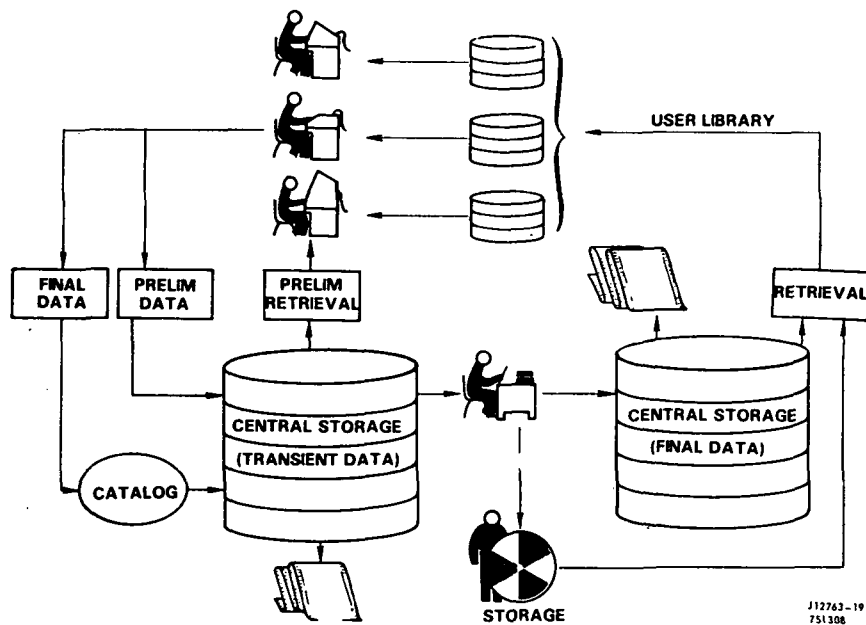


Figure 14.- Engineering data flow for computer-aided design and manufacture.

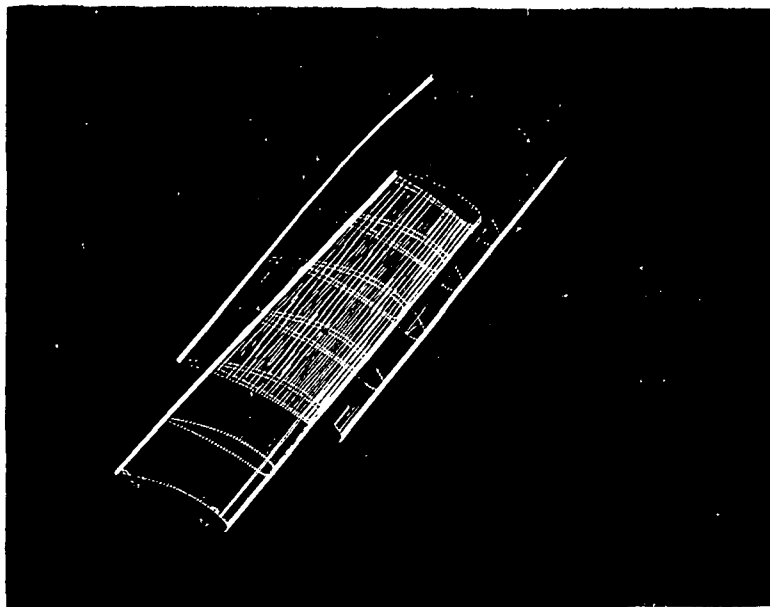


Figure 15.- Turbine core with tube partially extended. High performance display.

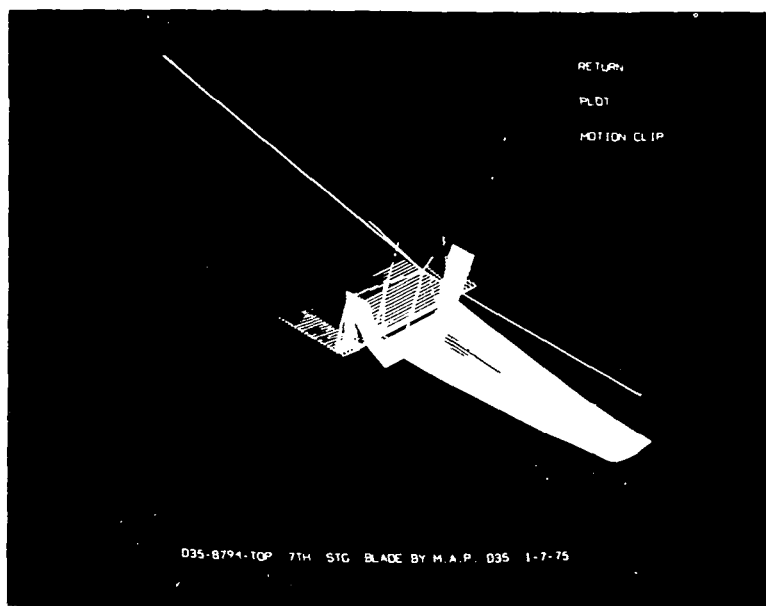
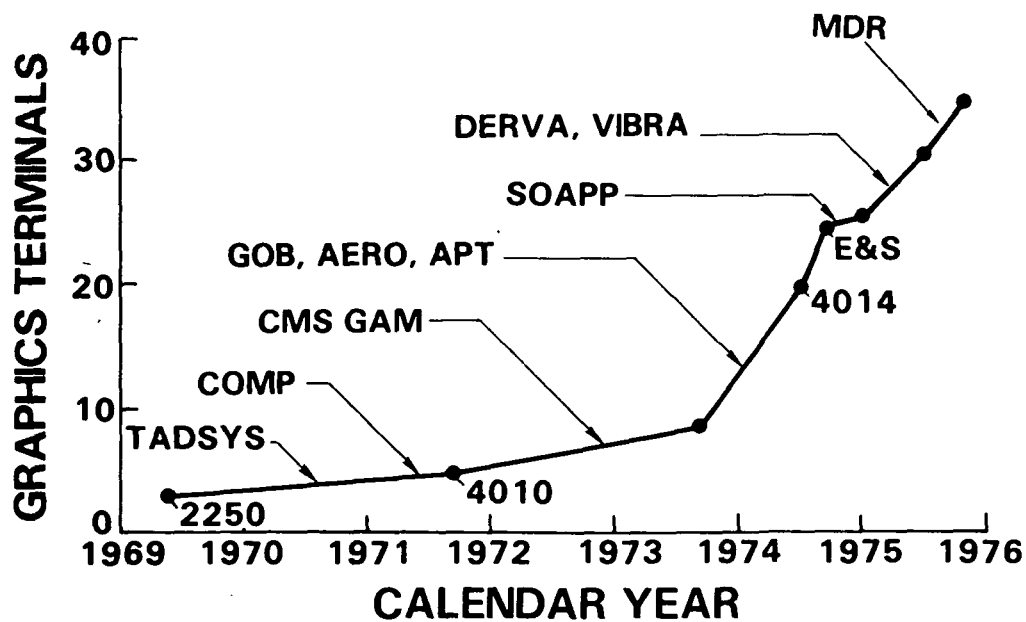


Figure 16.- Numerical control cutter path.  
High performance display.



J12763-20  
751308

Figure 17.- Interactive graphics growth.

**Page intentionally left blank**

# PACKAGING PRINTED CIRCUIT BOARDS -

## A PRODUCTION APPLICATION OF

### INTERACTIVE GRAPHICS

Walter A. Perrill

General Dynamics, Pomona Division

#### INTRODUCTION

Packaging electronic circuits onto printed circuit boards (PCB's) is becoming an increasingly difficult task. The conflicting requirements of "smarter", more complex circuits occupying less space have resulted in smaller PCB's containing more components. The increased use of integrated circuits and hybrid devices has only partially alleviated this situation. The situation is especially acute at General Dynamics, Pomona, because of the variety of circuits used. The design engineers package DC, analog, high speed digital and high frequency RF circuits - often in combination on the same PCB. Computer assistance is required to package these complex PCB's effectively.

Several batch mode programs for packaging PCB's were evaluated (starting in 1970) but none was satisfactory. The major reasons for this included the excessive amount of computer time required and unduly restrictive assumptions concerning component shape and number of conductor layers. Many of these difficulties seemed to stem from the fact that the programs attempted to achieve total automation of the packaging process with little or no human intervention. An analysis of the problems involved led to the belief that total automation was not likely in the near future.

Therefore, it was decided to try another approach using interactive graphics. This approach would allow the designer to monitor and direct the computer's actions in all phases of the packaging process. The intent was to combine the data storage and manipulation power of the computer with the imaginative, intuitive power of a human designer. This approach resulted in a computer program called the Interactive Graphics Packaging Program (IGPP), which is now in production use. This paper describes the structure and use of the IGPP.



## OVERVIEW OF HARDWARE AND SOFTWARE

The hardware currently in use at General Dynamics, Pomona, includes a CDC 6400 with 65,000 words of main memory and 125,000 words of extended core storage. A 30 inch Calcomp drum plotter is on line. The interactive graphics system (IGS) equipment consists of two CDC 777 terminals with 20 inch CRT screens, light pens, and keyboards (see Figure 1). The IGPP program is written in FORTRAN IV Extended with the exception of a few functions coded in COMPASS (assembly language). It executes in a field length of about 32,000 words. Each major function is implemented as a separate primary overlay to reduce the field length. The overlays reside on a random file in absolute code to reduce the time needed to locate and load the desired overlay. This improves system response to user commands.

The data base is maintained on a random disc file by an automatic data management system. This system partitions the data base into blocks and maintains the most frequently used blocks in a buffer in main memory. If the requested data is not in memory, the system automatically pulls the proper block into the buffer. If the buffer is full the least used block is written to the disc to make room for the new block.

The individual pieces of data are chained together in both random and sequential fashion. This structure permits rapid access to any piece of data as well as almost unlimited growth of the data base. The program continuously updates the data base so that it always reflects the current configuration of the PCB.

## PROGRAM OPERATION

The IGPP performs four major functions for the designer:

- (1) Data input and display
- (2) Component placement (either automatic or manual)
- (3) Conductor path routing (either automatic or manual)
- (4) Data output

In addition, the program contains built-in functions for detecting system and operator errors, debugging, and insuring the integrity of the data base. Any of these functions may be entered by the operator at any time.

## Data Coding and Entry

The initial action by the operator (i.e., the package design engineer) is to code the physical and electrical description of the circuit in a predefined data format. This includes the size and shape of the PCB, the basic grid size (typically .050 or .025 inches) and the window grid size (typically .5 or 1. inches). The window grid partitions the board surface into a set of squares for use in selecting "windows" to view areas of the PCB separately. Specific locations may or may not be assigned to each component. In addition, components may be locked into a fixed location, making them immovable even through operator error. The electrical schematic data is represented by assigning a unique number to each electrically distinct signal net. This number is associated with each component pin which is electrically common to the net. The data is punched into machine cards which are read by the program.

## Data Display

The designer has the option of displaying the entire PCB, or selecting any rectangular subset of the window grid squares. The positions of the window grid squares are indicated by the letters A, B, C, D, ... displayed at the upper and right edges of the PCB. The designer selects the four letters which define the left, right, upper and lower boundaries of the desired area.

The components are displayed at their assigned locations on the PCB. Components without pre-assigned locations are automatically placed at the top of the screen (see Figure 2). This is done to avoid prejudicing the designers initial judgement.

## Component Placement

The designer may select either of two modes of component placement, automatic or manual. In the automatic placement mode the computer uses a center of mass algorithm to determine the component locations. The designer may elect to have all of the components placed at once over the entire PCB surface, or he may select up to sixteen components to be placed into some specified area. The algorithm groups the components into electrically related families but does not attempt to rotate or move components to eliminate overlaps (see Figure 3). After the automatic mode is complete, the designer enters the manual placement mode to make the necessary adjustments.

In the manual mode the electrical schematic data is indicated by displaying a set of lines, called vectors, from the pins of the component being moved to all other electrically common pins (see Figure 4). The designer may display vectors from any or all of the pins on the selected component.

To move a component the designer selects it with the light pen, then moves the pen with the component following. Components may be rotated as well as translated, giving the designer complete control over component placement. When the component is released the program automatically rounds its position to the basic grid of the PCB. This process continues until the designer is satisfied (or nearly so) with the placement of all of the components (see Figure 5).

### Conductor Path Routing

The conductor path routing function may be used in either the automatic or manual mode. In the automatic mode the computer uses a row/column algorithm to route the conductor paths. This algorithm is related to the concept of a channel router in which the width of the channels is equal to the basic grid of the PCB. The channels are horizontal on the front of the PCB (rows) and vertical on the rear (columns). Each channel may contain, at most, one conductor at any given point. Conductors are routed following the channels although short sideways jogs (up to two or three channels) are permitted when dodging small obstacles. Longer sideways runs are completed by making a via (a drilled hole plated with conductive solder) to the other side of the PCB and finishing the run there. The designer may elect to route the entire PCB or only a selected area. When the automatic routing function is complete (see Figure 6) the designer may enter the manual routing mode to make any necessary corrections.

In the manual mode each electrically distinct signal net is routed individually, with the designer selecting the net to be routed. Electrical schematic data is indicated by flashing dots in all component pins which are common to the selected net; electrically common conductors are also flashed. The conductor paths are routed by "drawing" them on the IGS screen with the light pen (see Figure 7). The designer may select the path width and layer of the conductor being routed; up to four layers of conductors are permitted. On the IGS screen the layer of each line is indicated by displaying the line in a different style (solid, dashed, or dotted). Lines may be changed from one layer to another or erased completely. Other options automatically round conductor positions onto the basic grid and align paths horizontal or vertical. Dimension checking options are also available. . Due to hardware limitations on the size of the display file (approximately 7500 16-bit words) only the center line of each path is displayed. However, the path width is indicated during routing by a small circle displayed at the point of the light pen.

## Data Output

When the PCB design is complete the designer may select any of the following output options:

- (1) Pen plots showing component and conductor positions (see Figures 8 through 10). These plots are used both in verifying PCB producibility and in formal documentation.
- (2) Pen plot showing the size and location of each hole to be drilled. This plot is used in formal documentation and in the factory production process (see Figure 11).
- (3) Card deck for numerically controlled drilling machines. This deck, related to (2) above is used in the factory production process.
- (4) Card deck to drive a numerically controlled photoplotter (Gerber 1232). The photoplotter produces the artmaster which becomes part of the formal documentation and is used in the factory production process (see Figure 12).
- (5) Printed listing giving a summary of the entire program run.

## Data Base Protection

The safety of the data base is insured by the checkpoint function. The designer selects this function at regular intervals during the program run, writing a copy of the current data base to a permanent disc file. A checkpoint file is also written automatically upon program termination. This permits the designer to terminate the program run at any time without loss of data. At some later time the designer may make another program run, access the latest checkpoint file and continue working. In the event of machine failure only the changes made since the last checkpoint are lost; the job may be restarted and continued from that point. For long term storage the data base may be saved on magnetic tape. In the event of later design changes the data base may be restored to the disc and the IGPP used to make the necessary changes.

## Error Control

Minor debugging functions include printing the last ten light pen selections and the last five data base references, data base dumps and field length core dumps. These functions may be selected manually by the designer, or automatically by the program when certain error conditions are detected.

## CURRENT DEVELOPMENT EFFORTS

The IGPP program is constantly being modified to expand its capabilities, and to meet changing requirements. Among the improvements currently under development are the following:

- (1) Automatic checking of conductor clearance and other design rules.
- (2) Display warning messages to the designer in case of violations.
- (3) Optimization of plotter data. This includes automatic fill-in of ground plane areas and reduction of wasted plotter motion.
- (4) Gate and lead swapping to allow the designer more freedom in digital circuits.

Although the IGPP was developed for packaging PCB's, it was designed to be as general as possible. This generality has permitted the use of the IGPP as a basis for new systems. The development time for new systems is being reduced since many of the display, data base and output functions already exist in the IGPP.

One version is being developed to package integrated circuits and hybrid devices (see Figure 13 and 14). This version contains functions to allow for jumper wire bonds (flying leads), thick film resistors, predefined conductor patterns and automatic checking of critical design rules.

In another project the IGPP is being merged with a set of programs which design passive microwave stripline and microstrip components. Used by a microwave engineer this program will accept electrical data (frequency, dielectric constant, etc.), design a selected component (filter, coupler, etc.) and store it in the data base (see Figure 15). The engineer may recall the stored components, position them on the substrate, and connect them into a completed circuit (see Figure 16). The program will generate the data to produce the artmaster and all other required documentation.

In another project the IGPP was interfaced with a program which simulated a naval weapons exercise. Ships and aircraft were displayed on the screen along with data on their position and motion. Additional data were displayed when the vehicles were selected with the light pen. The exercise planners were able to visually monitor the progress of the exercise and detect problem areas easily. Calcomp pen plots of each important event were included in the formal documentation.

## CONCLUSION

The IGPP has been used in production applications since 1972. During this time it has been used to package over 130 PCB's containing an average of 50-75 components and two layers of conductors. Electrically the circuits were primarily analog and RF although digital circuits account for an increasing portion of the total. The most complex PCB packaged to date measured 16.5 cm (6.5 in.) by 19 cm (7.5 in.) and contained 380 components, two layers of ground planes and four layers of conductors mixed with ground planes. Electrically, the circuit contained RF signals extending into the microwave region, high speed digital signals and DC control signals. Conductor paths included stripline buried between ground planes as well as normal conductors. The IGPP produced the basic documentation for the board although the photoplotter data required a good deal of manual manipulation because of the extensive ground planes. The time required to package the board was just over four weeks, from schematic diagram to a finished set of artmasters. Figures 17 and 18 are reproductions of artmasters for two of the six layers.

We feel that the IGPP is a proven production tool now and has great potential for growth. We intend to improve its capabilities and expand its use until it is the main tool for packaging PCB's at General Dynamics.



Figure 1. - Design engineer seated at the graphics terminal. The display screen, keyboard and light pen are the interactive elements of the system.

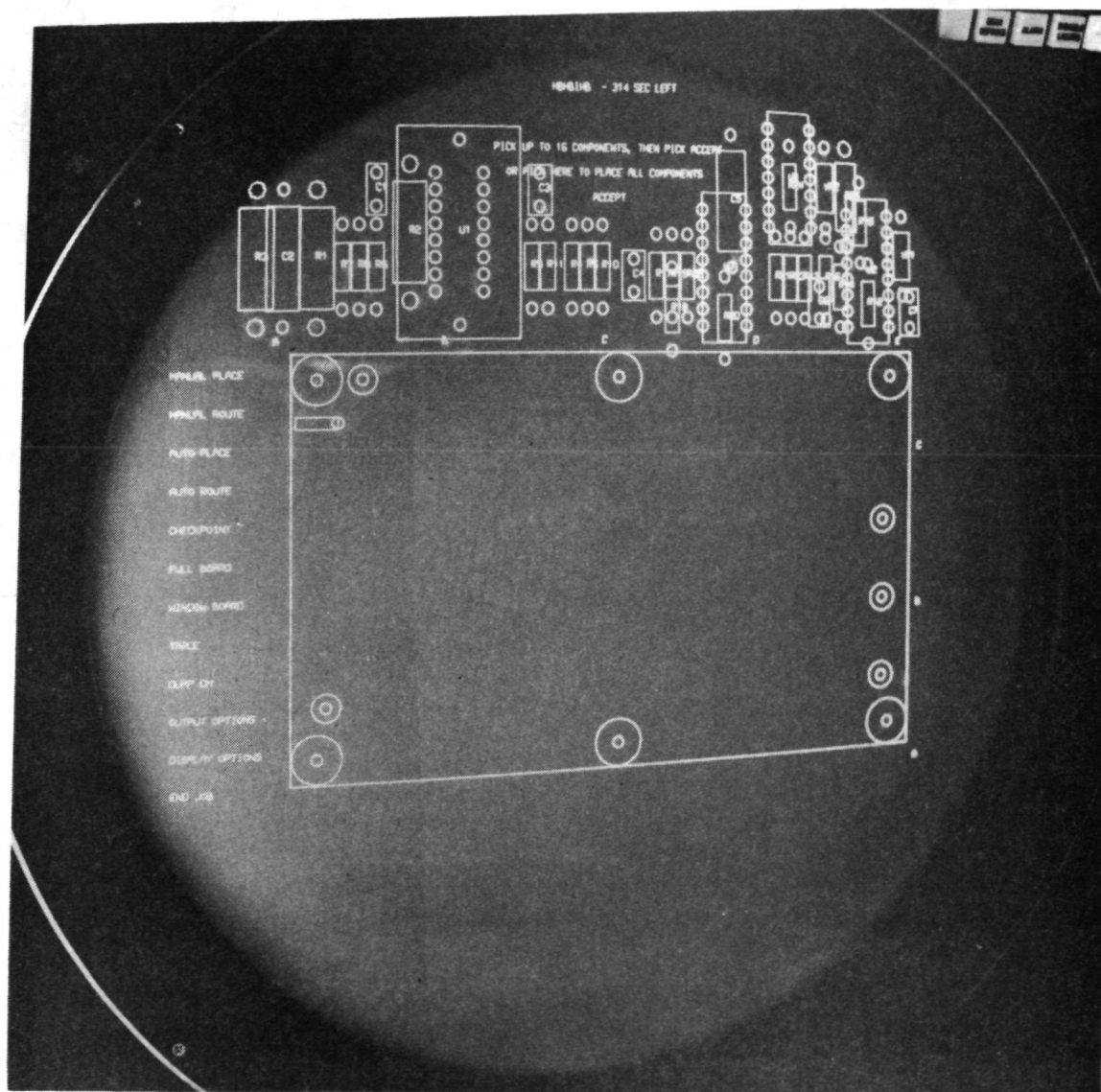


Figure 2. - Display of printed circuit board and components. Prior to the start of packaging, the components are positioned at the top of the screen. The text at the top of the screen indicates the start of the automatic component placement function.



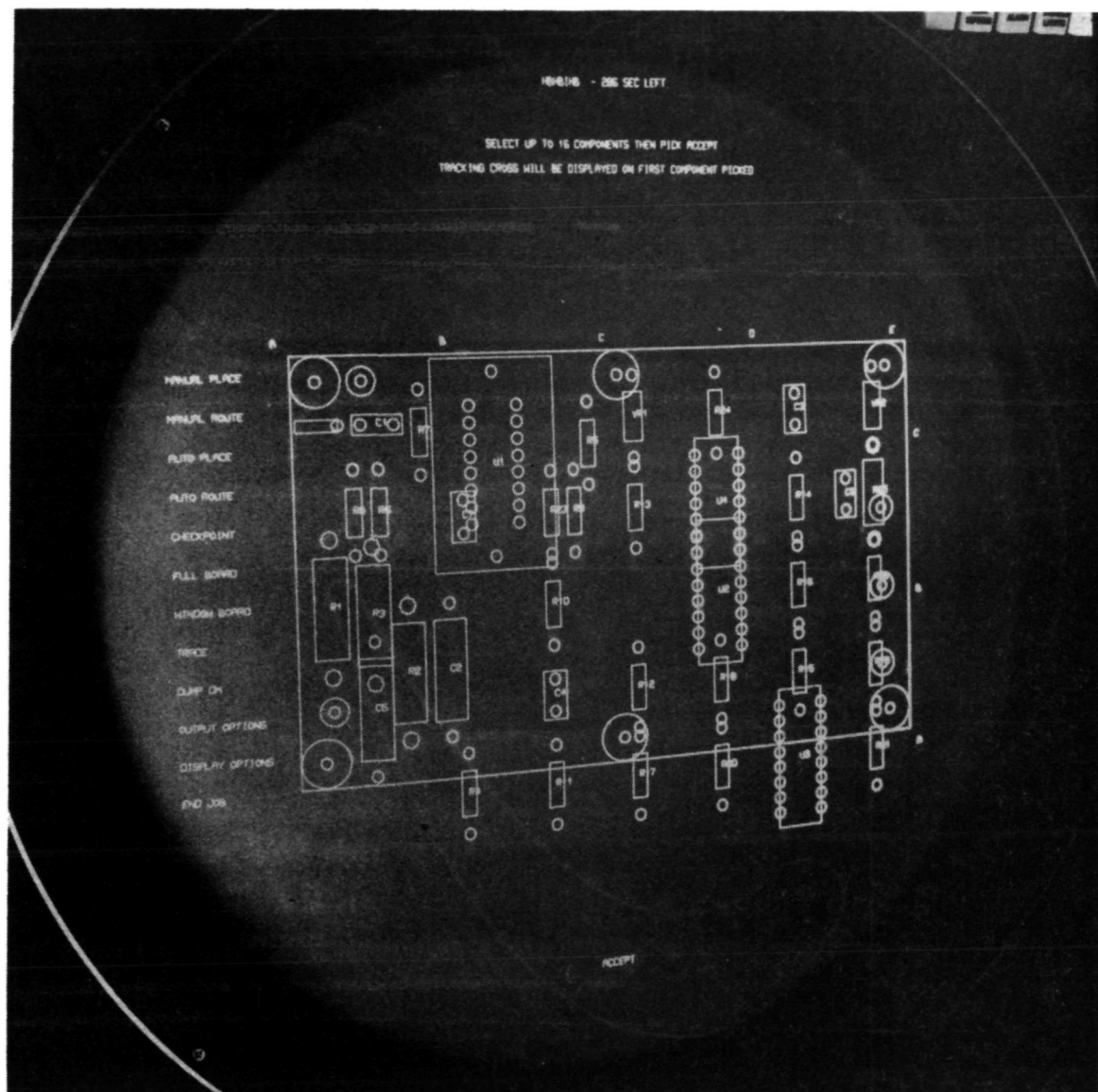


Figure 3. - PCB after completion of automatic component placement. The placement algorithm groups the components into electrically related families but does not optimize the placement to eliminate component overlap. This placement required about 10 seconds of computer time.

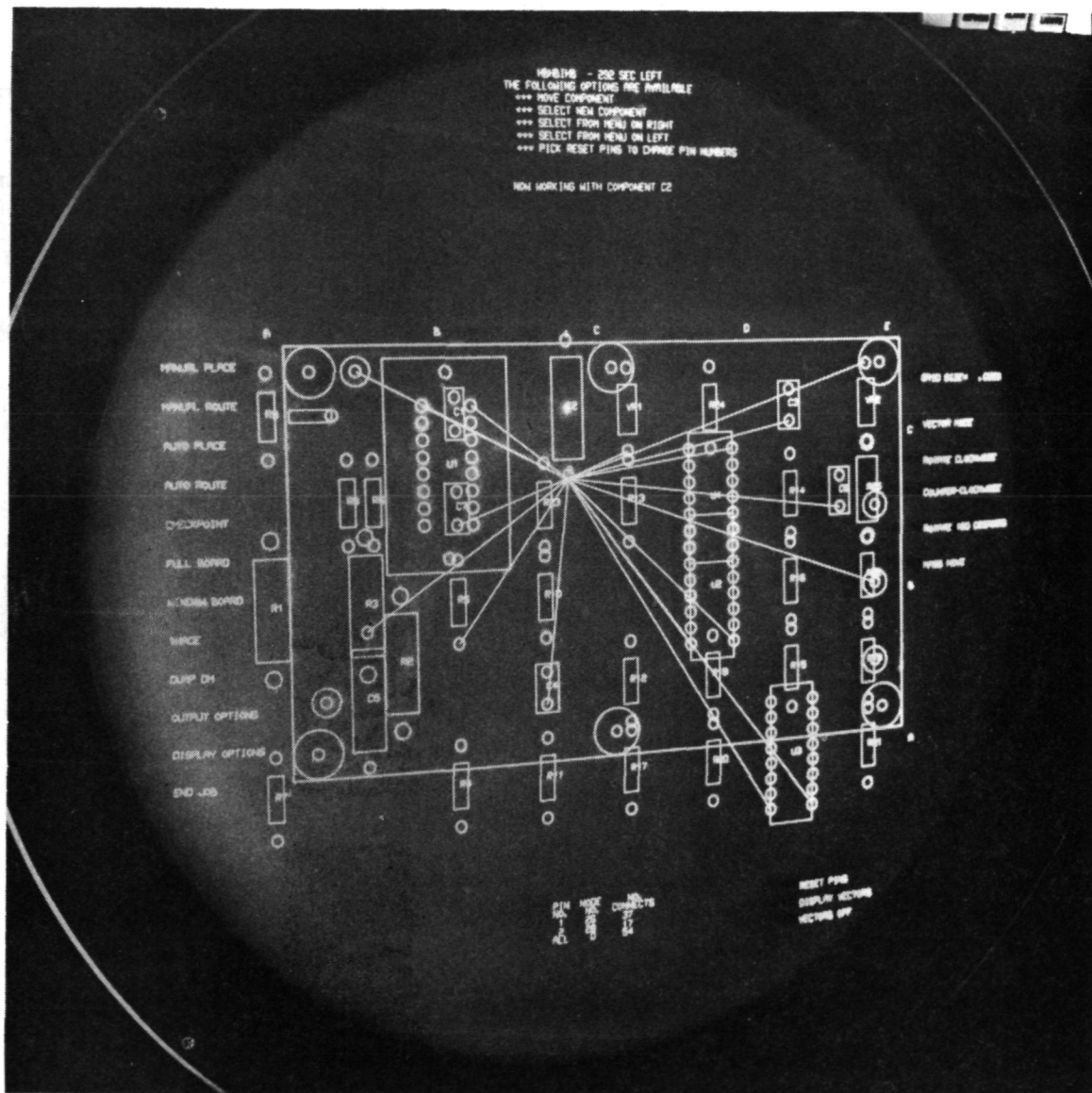


Figure 4. - Vector display during manual component placement.  
 Vectors are displayed from pin 2 of component C2 (the  
 one being moved) to all other electrically common pads.

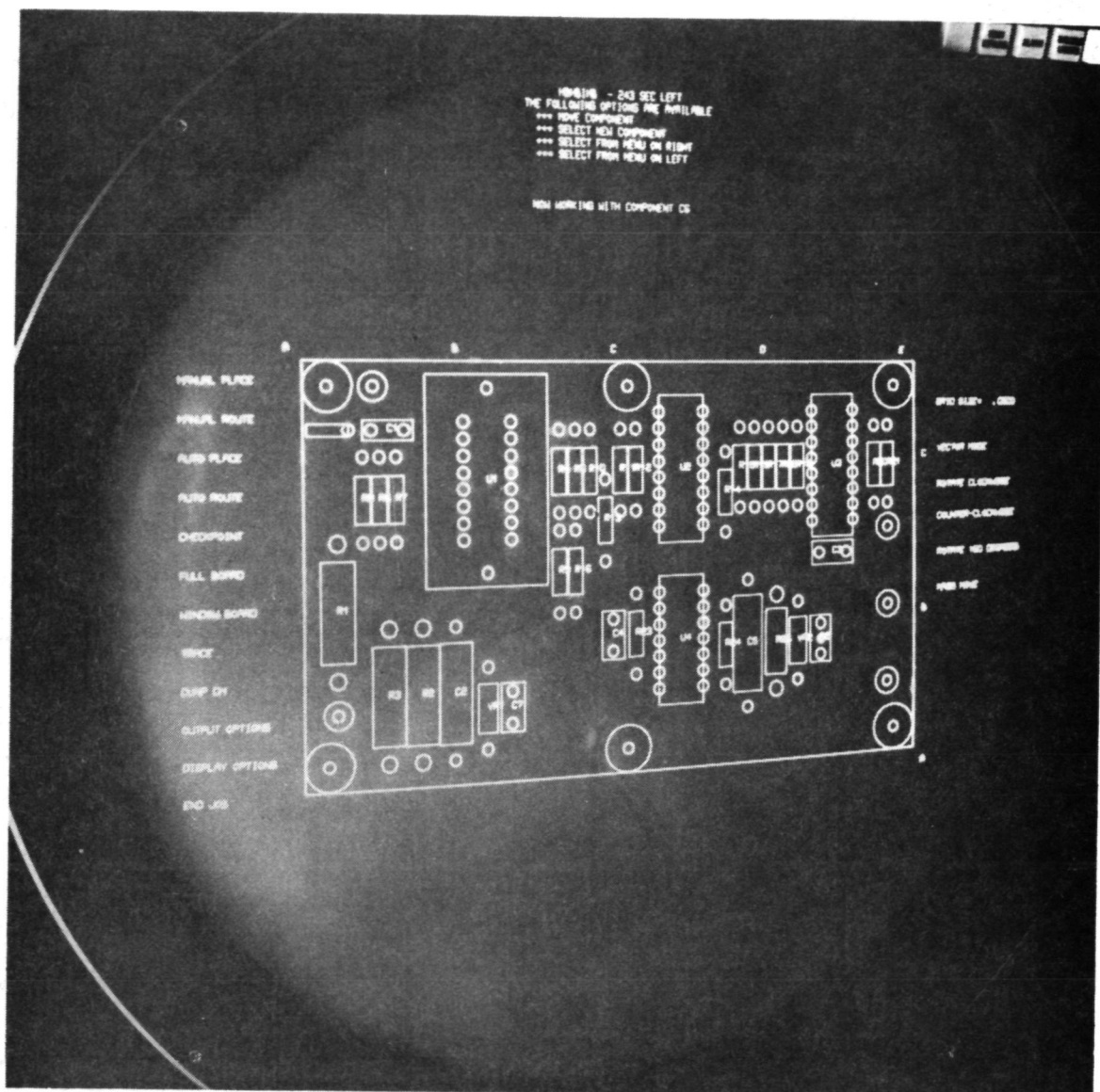


Figure 5. - Final component placement after completion of the manual placement function.

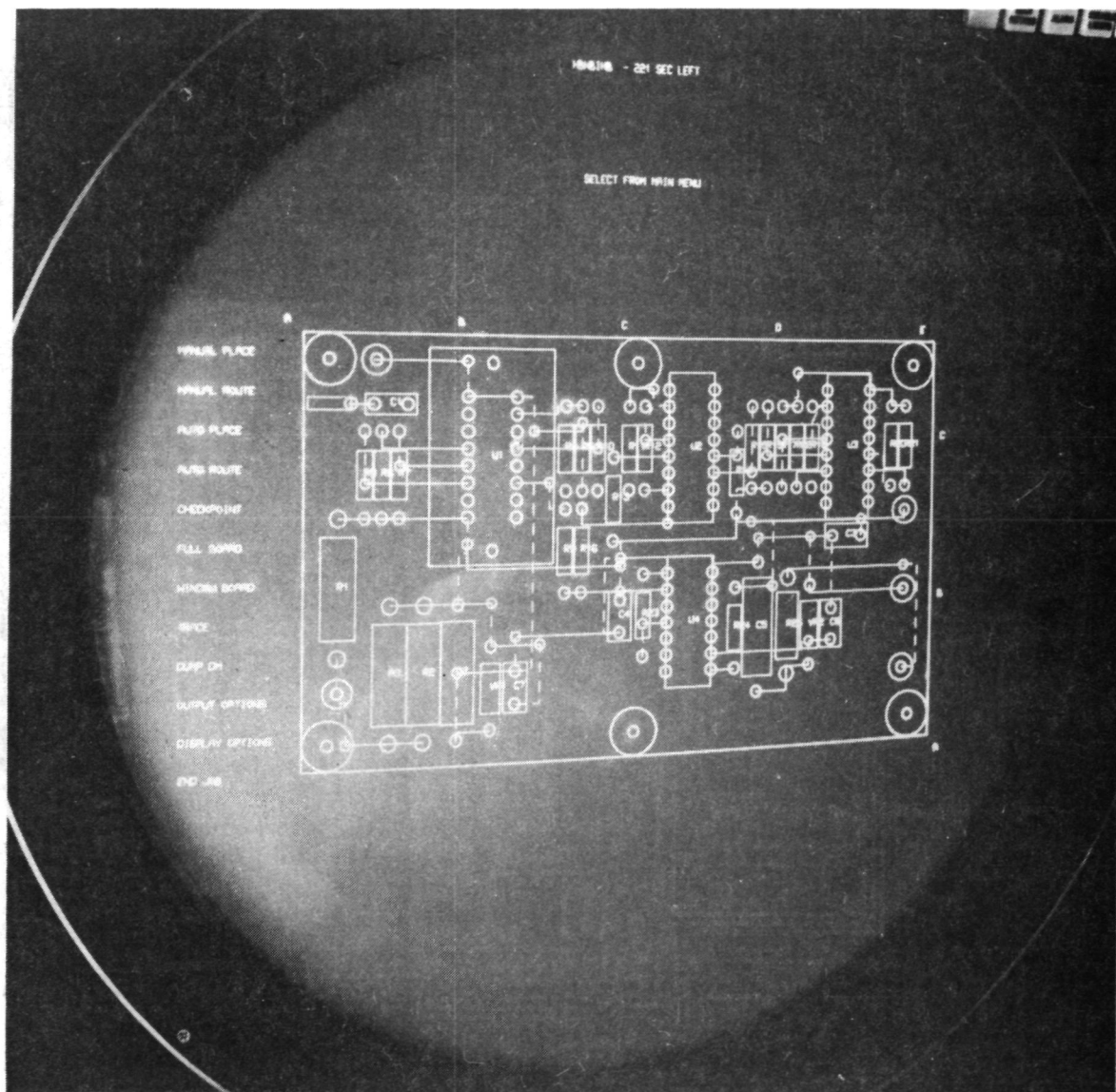


Figure 6. - PCB after completion of the automatic conductor path routing function. The solid paths are on the front side of the board and the dashed paths are on the rear. This route required about 15 seconds of computer time.

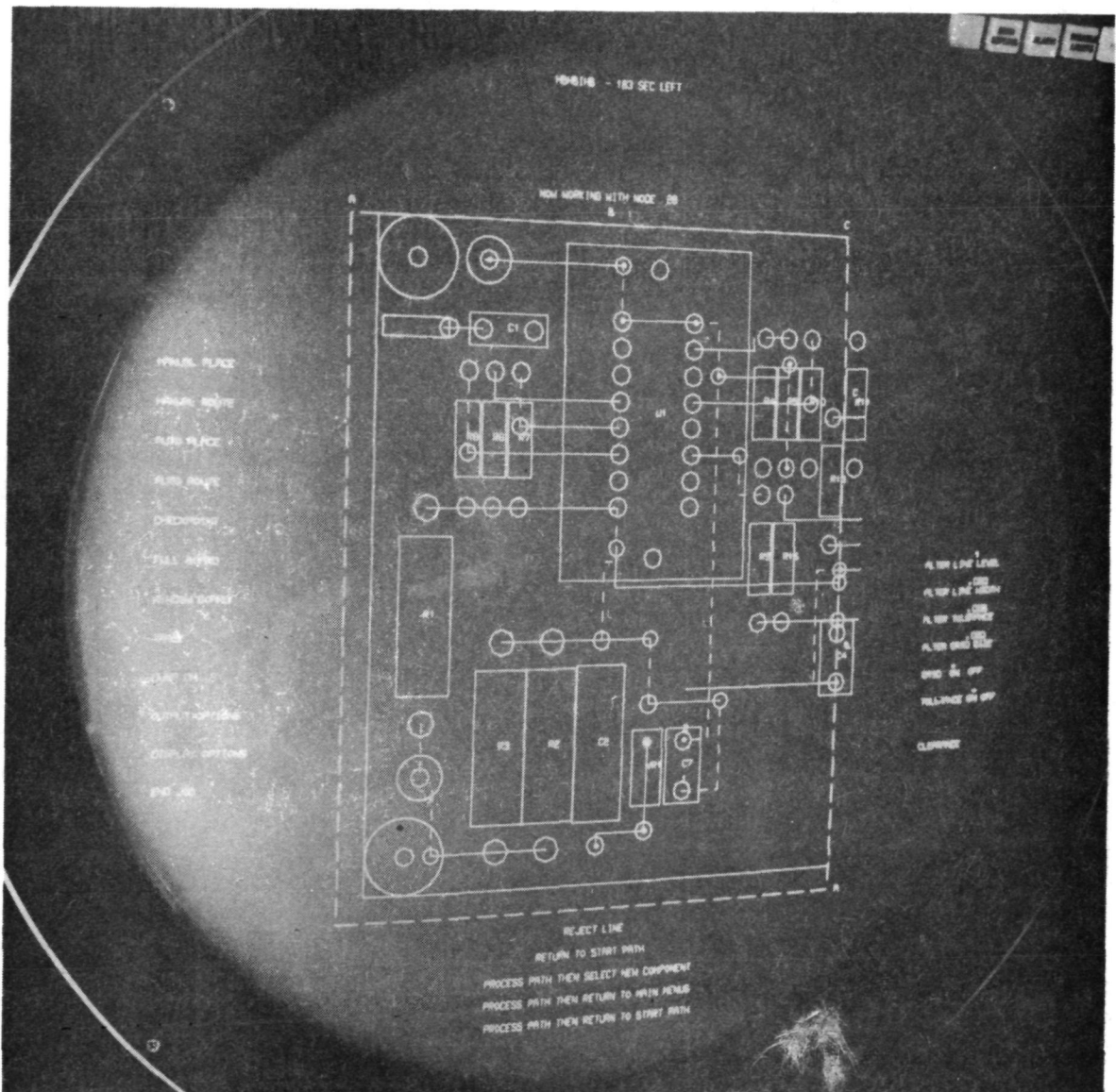


Figure 7. - Window view of the left half of the PCB during the manual conductor routing function. Bright dots appear in conductor pads which are electrically common to the signal net being routed. The conductor path terminating in a bright spot (the tracking symbol) in the lower center of the screen is the path being routed.

LAYER NO. 1

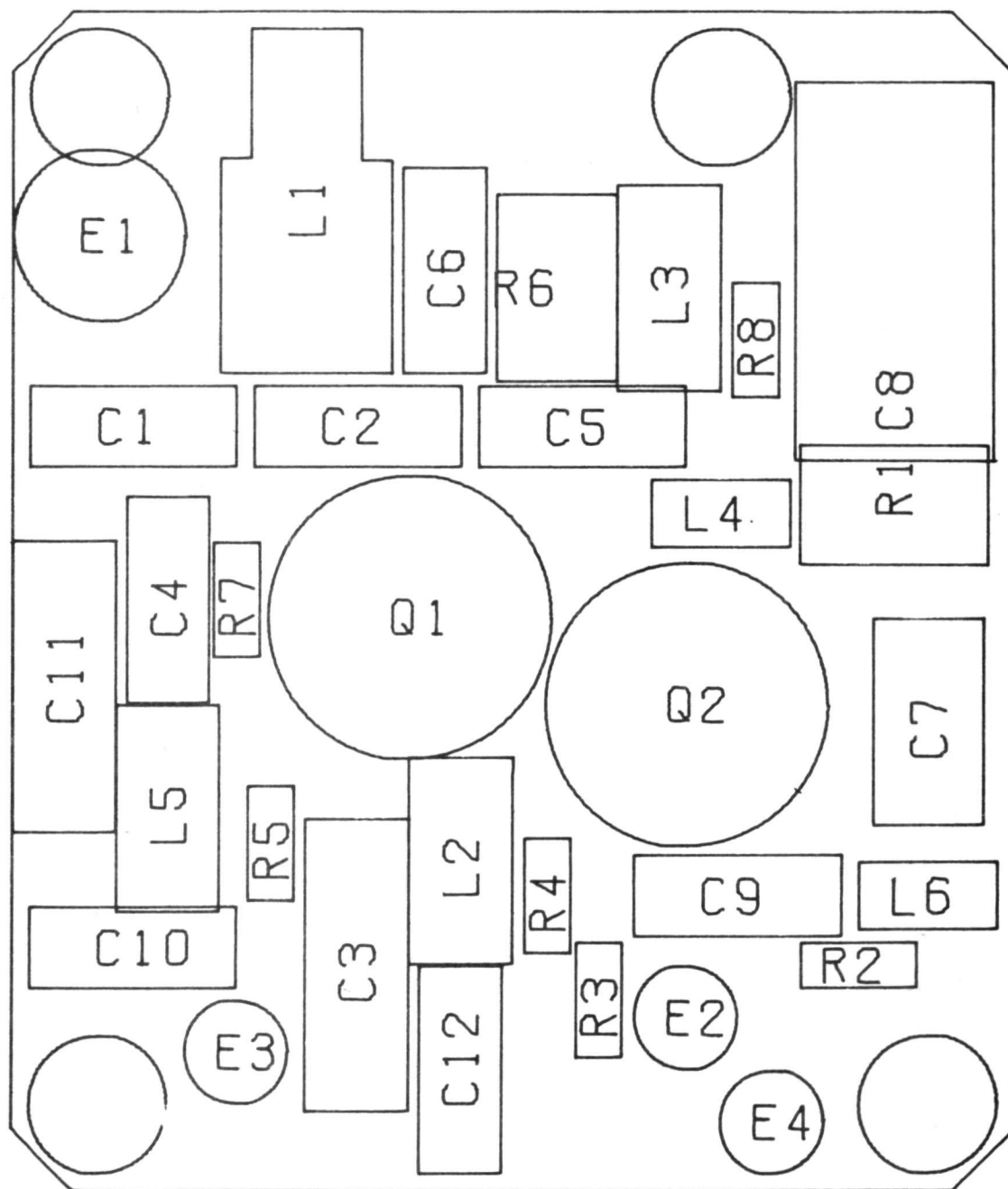


Figure 8. - Calcomp pen plot of a small RF board, 4 times actual size. This is the assembly plot showing the position of each component on the finished board.



LAYER NO. 1

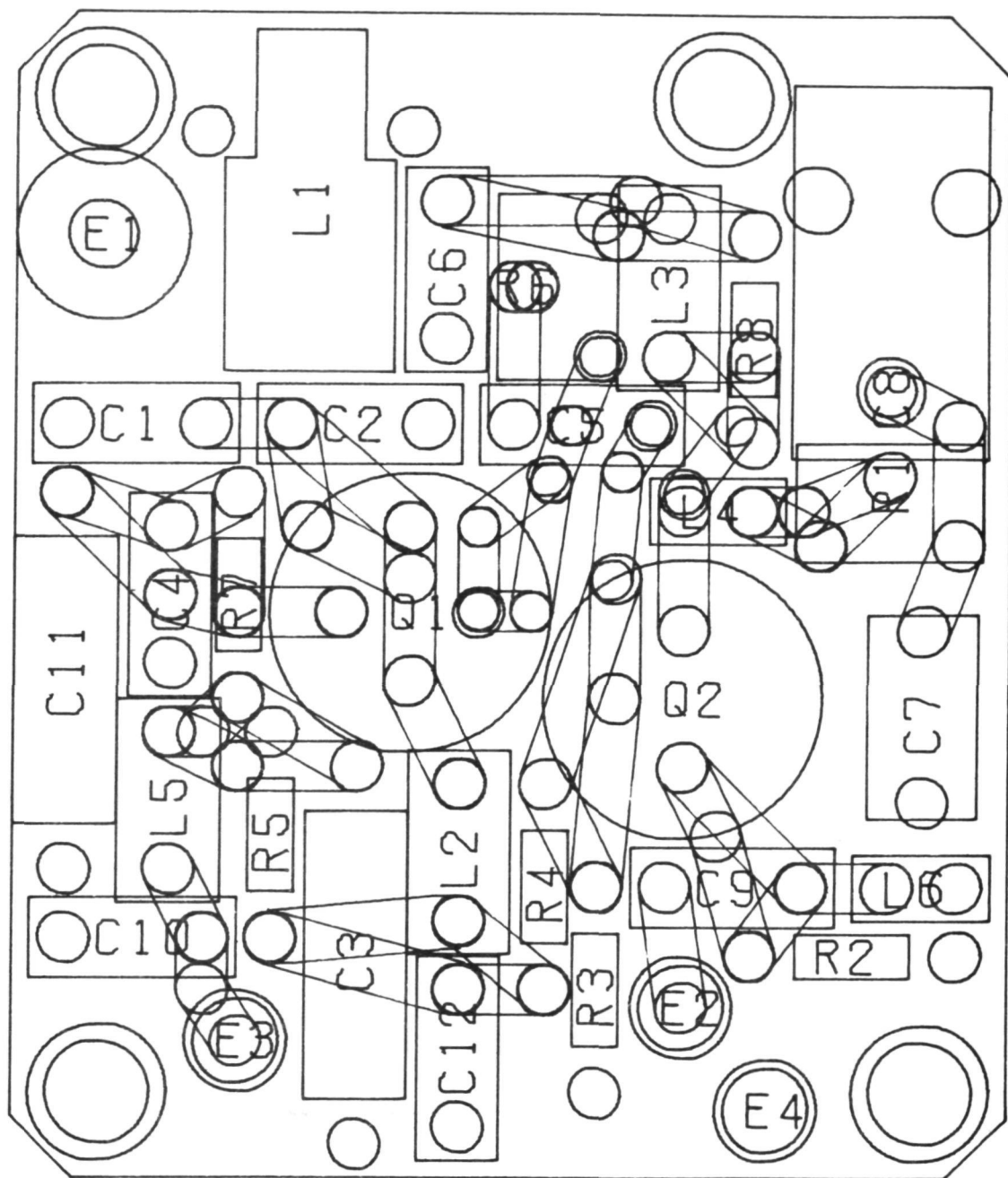


Figure 9. - Calcomp pen plot of the PCB of Figure 8. This is a check plot showing the components, their connection pads and the conductor paths on the front side of the board (layer no. 1). The conductors are plotted showing both edges of the path and circles at the end and turning points. This approximates the appearance of the finished artmaster, since the conductors are drawn with a round light beam.

LAYER NO. 3

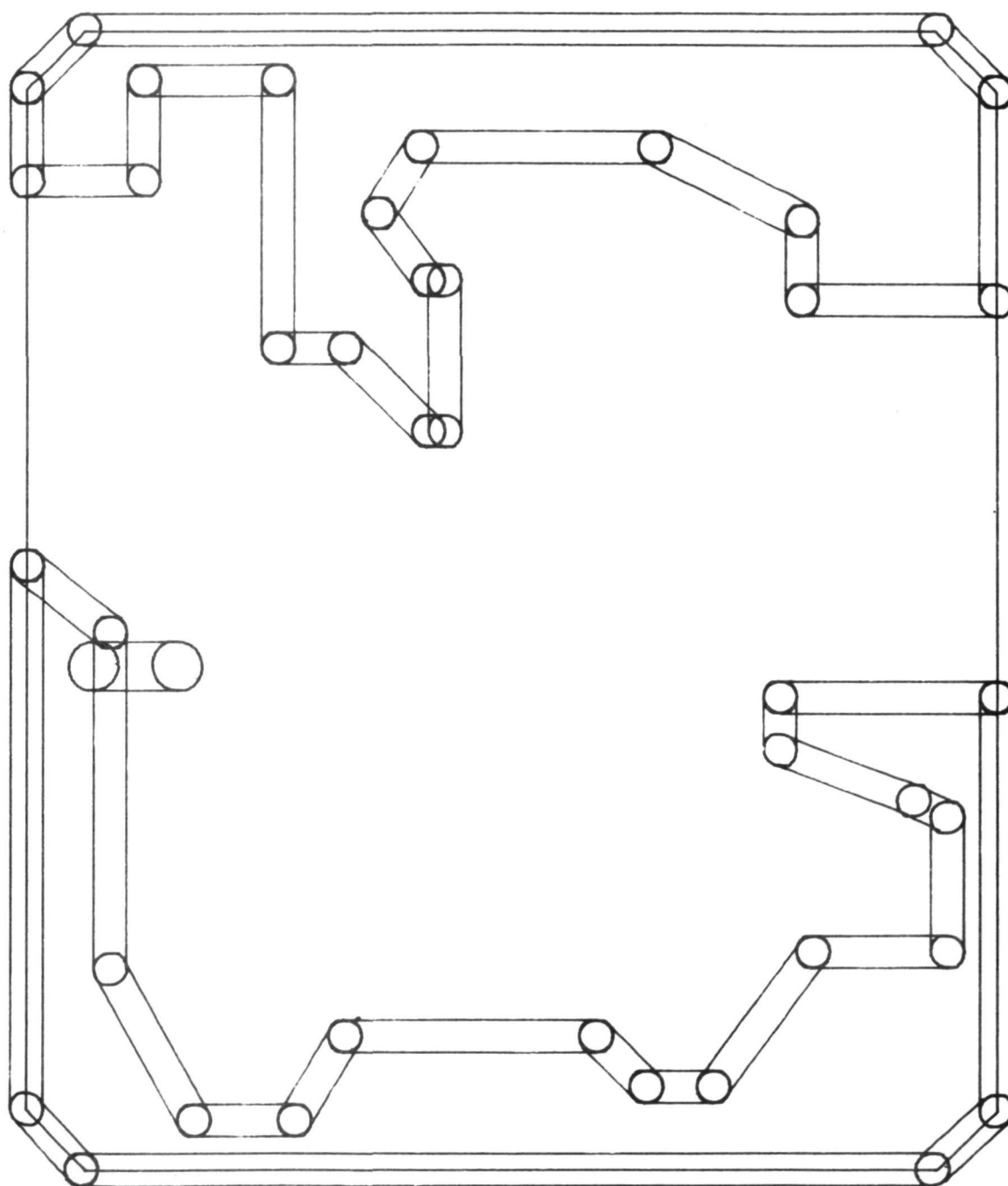


Figure 10. - Calcomp pen plot of the PCB of Figure 8. These conductors serve to outline areas which will become ground planes (areas of metal electrically connected to ground) on the finished board.



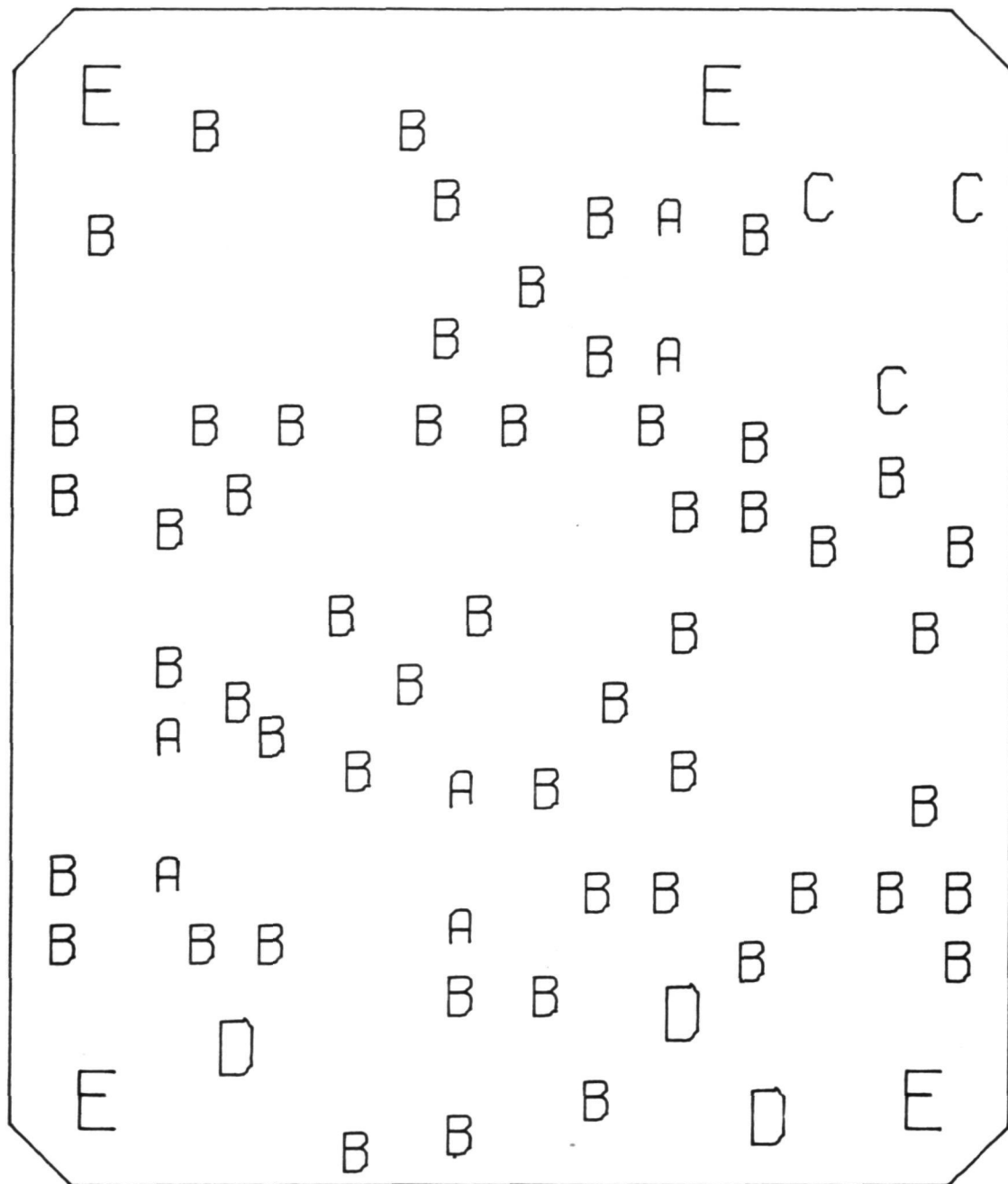


Figure 11. - Calcomp pen plot of the PCB of Figure 8. The letters indicate the size and location of each hole to be drilled.

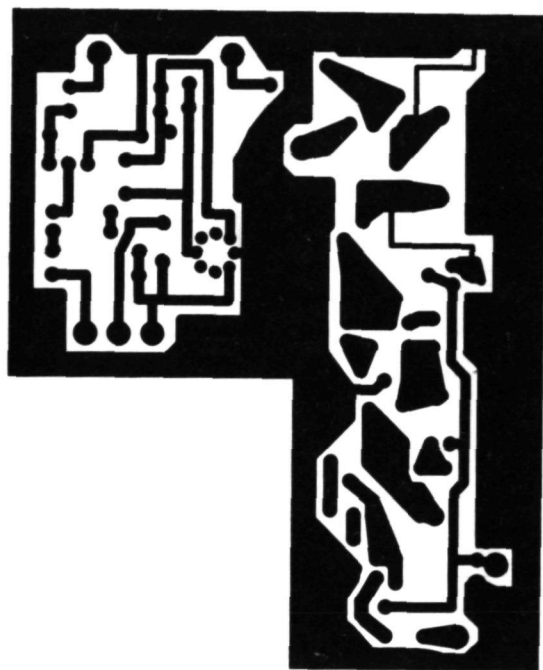


Figure 12. - Artmaster for another RF board, shown actual size. The dark areas represent areas of metal on the finished board. This artmaster (as well as Figures 17 and 18) was produced by a Gerber 1232 photoplotter.

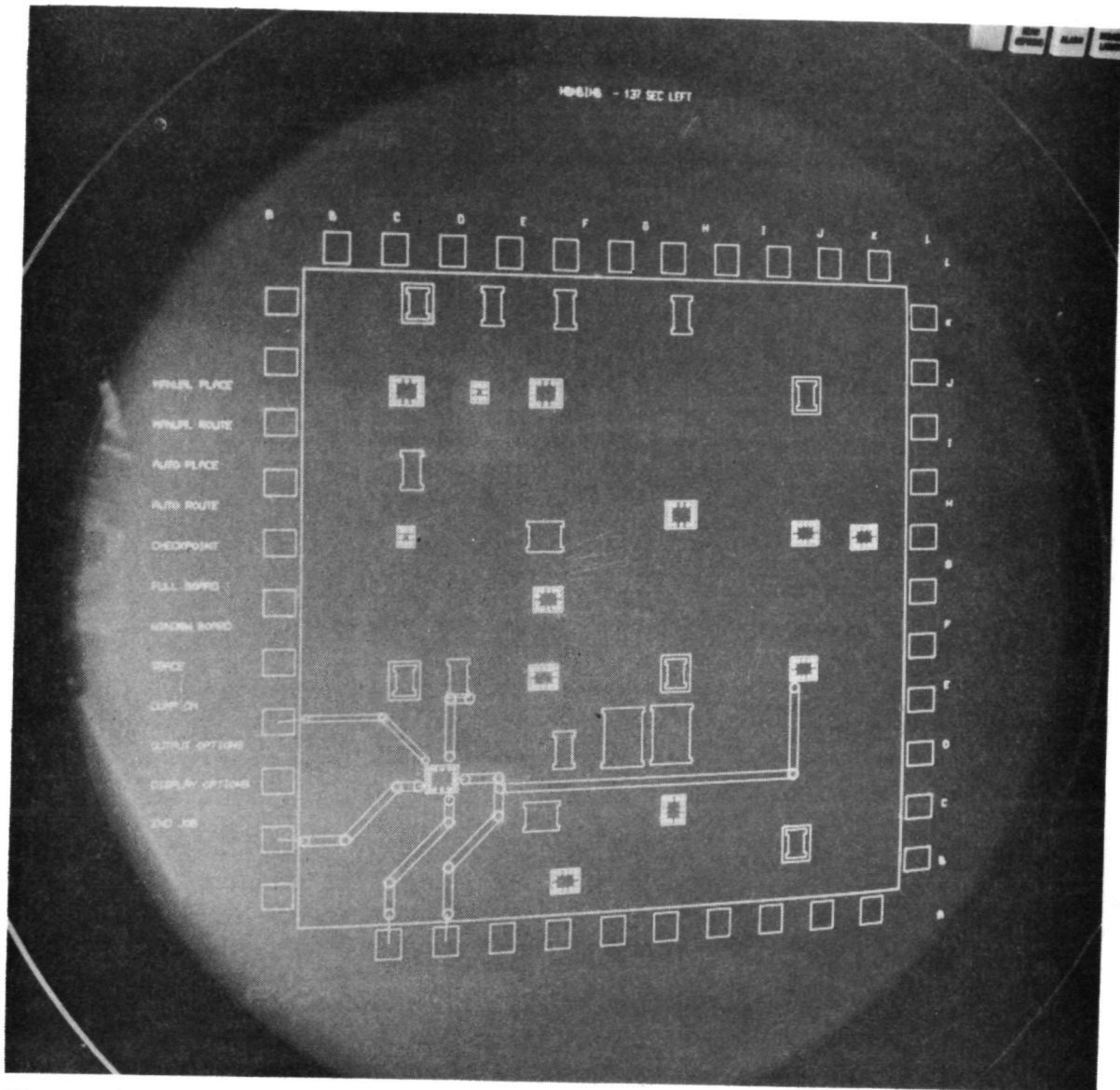


Figure 13. - Display of a hybrid microelectronic device. The components displayed represent integrated circuit chips and thick film resistors. The conductors illustrated include jumper wires (the short, straight lines) and conductors on the substrate. The actual size of the substrate is 2.54 cm (1 in.) square.

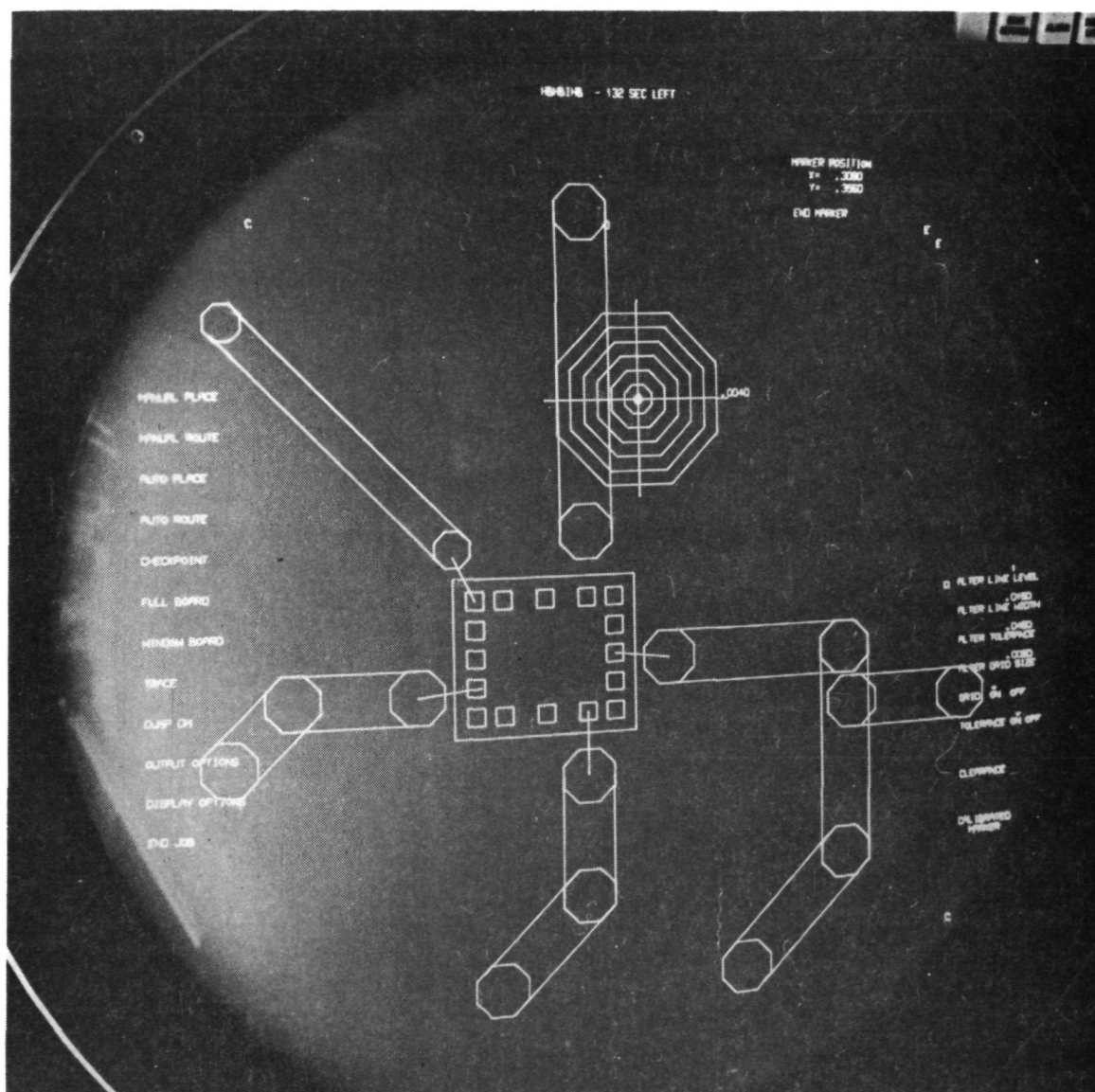


Figure 14. - Window view of lower left corner of the hybrid substrate showing a single integrated circuit (IC) chip. Jumper leads are shown connecting the substrate conductors to the bonding pads on the upper surface of the IC chip. Also shown is the calibrated marker (concentric octagons) for measuring critical dimensions.

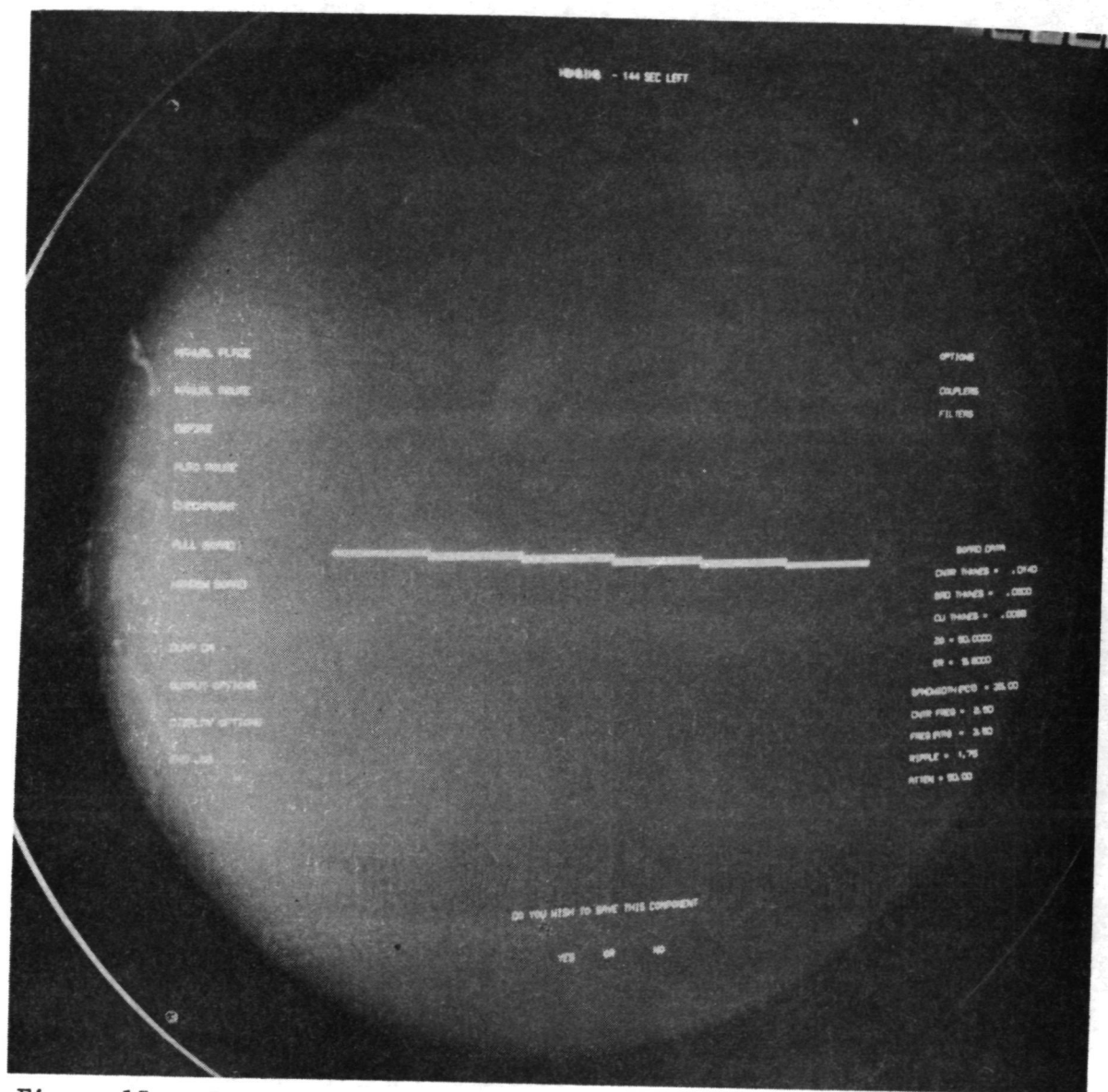


Figure 15. - Display of the microwave stripline program in the component design mode. The performance parameters entered by the designer are listed at the right side of the screen. The parallel line coupled filter displayed at the center was designed to meet those parameters.



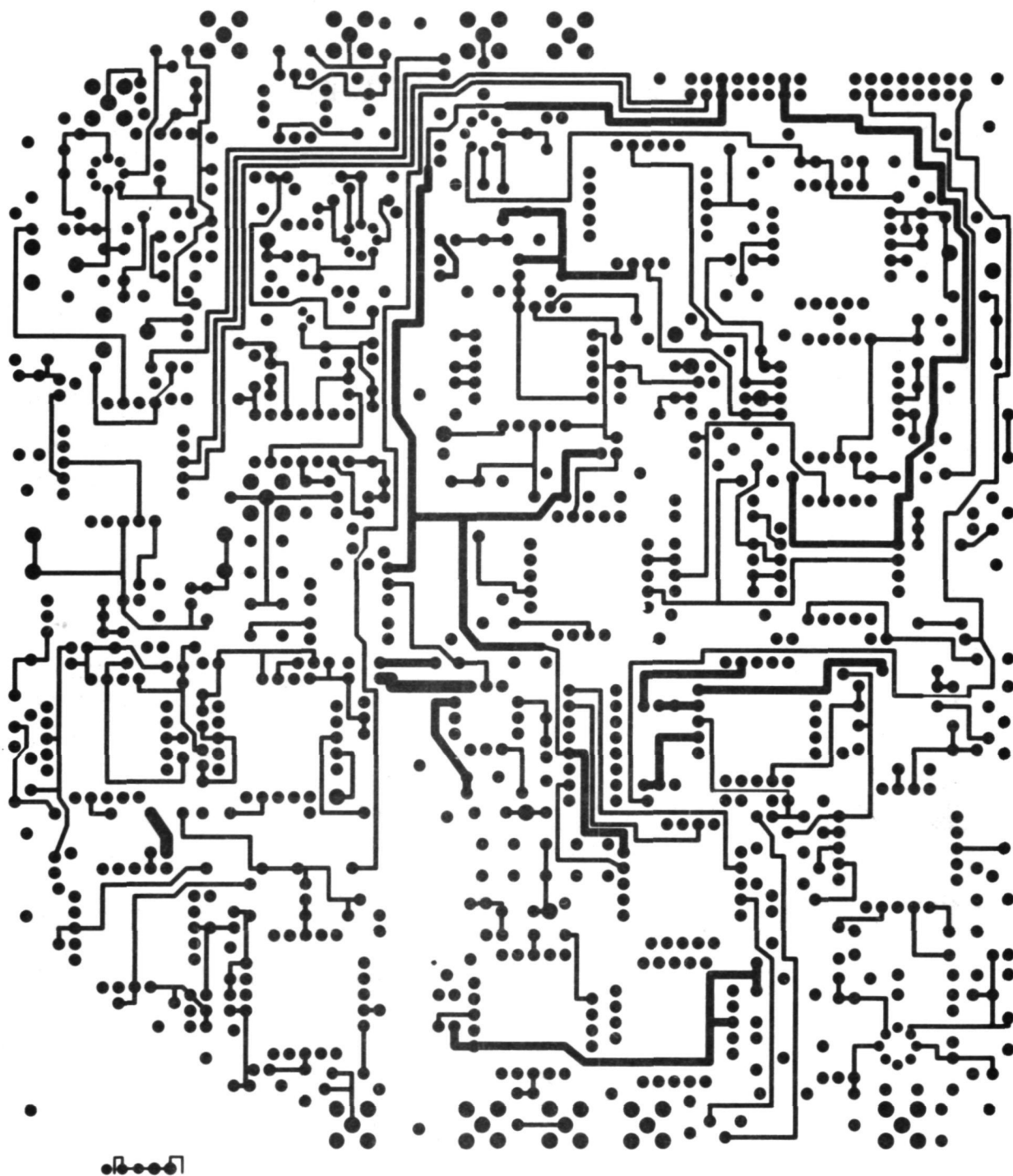


Figure 17. - Artmaster for one layer of a complex PCB, shown actual size.  
This layer contains only component pads and conductors.



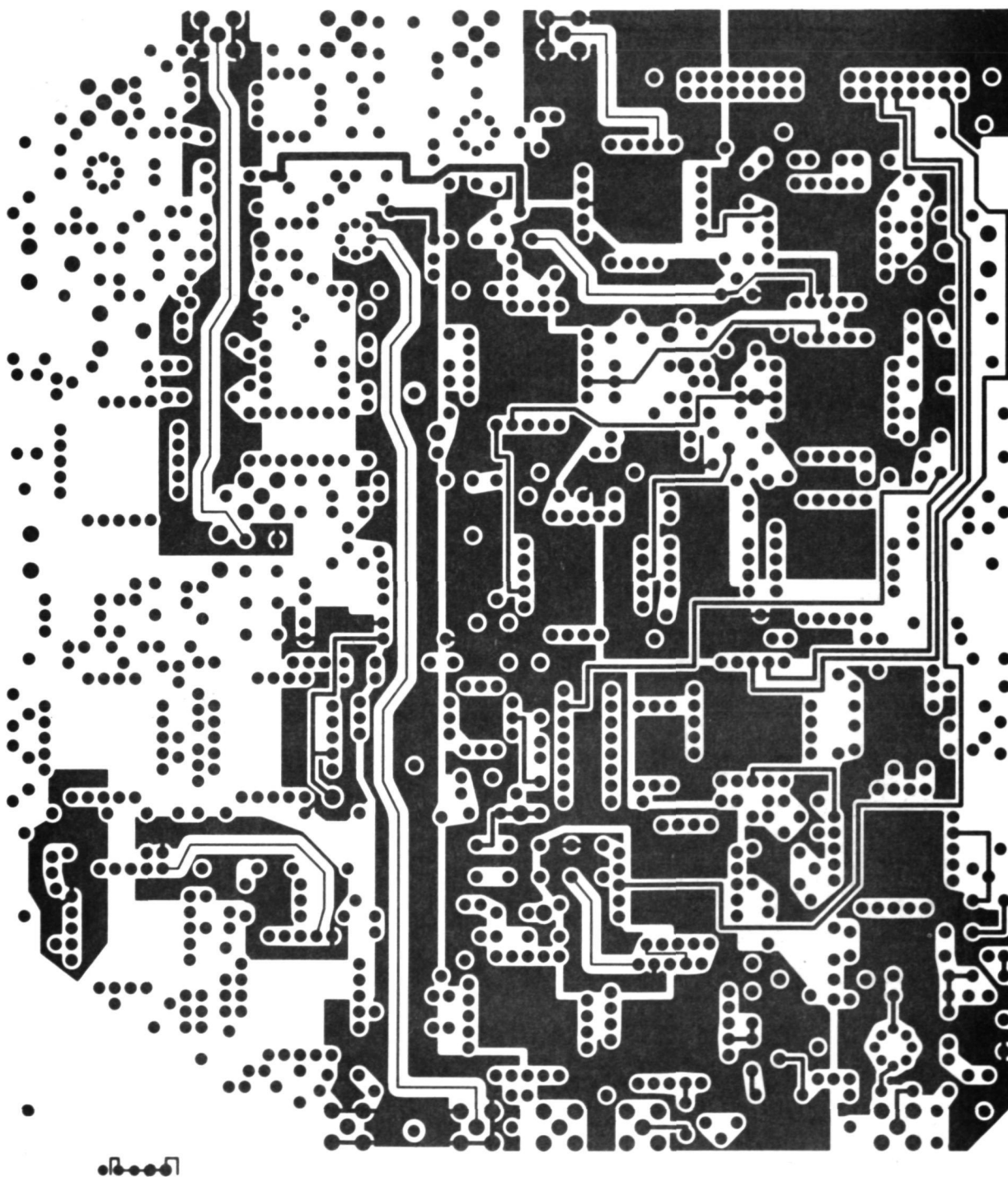


Figure 18. - Artmaster for another layer of the PCB of Figure 17. This layer contains extensive areas of ground planes.



Page Intentionally Left Blank

DESIGN AND IMPLEMENTATION OF A MEDIUM SPEED COMMUNICATIONS  
INTERFACE AND PROTOCOL FOR A LOW COST, REFRESHED DISPLAY COMPUTER

By James R. Rhyne and Mart D. Nelson

The University of Houston  
Department of Computer Science

INTRODUCTION

Refreshed displays have several advantages over storage tube displays, but the primary advantage is the ability to selectively delete or alter portions of a display without having to re-draw the entire display [1]\*. When such displays are used in conjunction with a large scale computer system, interaction with a computer program has new dimension of ease and flexibility. Because such displays must re-draw the entire picture several times each second to prevent it from flickering or fading from view, quite a bit of processor power can be tied up in the task of keeping the picture visible.

Some displays have a special processor, called a display processor, which refreshes the picture on the screen at regular intervals, thus removing a substantial processing load from the large scale computer system. It is possible to buy a display system with a built-in display processor for under \$20,000, and recent advances in LSI technology indicate that more such display systems will be available in the future. Most of these low cost displays have been used either as stand-alone systems or have been connected with a large scale computer system by low speed, asynchronous communication lines.

DESIGN CRITERIA

We were curious to see what difficulties and benefits would result from using a 40,000 bit/second communication line as the connecting link between an IMLAC PDS 1-D display computer [2] owned by the Department of Computer Science and the University of Houston's Univac 1108 computer system. The obvious benefit is a relatively high speed for data transfers between the 1108 and the IMLAC which permits rapidly changing displays. The main difficulty lies in designing a hardware and software interface which would meet the design criterion of 40,000 bit/second communication without being expensive. This paper describes the hardware and software system which resulted.

Figure 1 shows a block diagram of the IMLAC. It consists of two independent processors which share a common memory. The display processor generates the deflection and beam control currents as it interprets a program contained in the memory; the minicomputer has a general instruction set and is responsible

---

\*Numbers in brackets indicate references.

for starting and stopping the display processor and for communicating with the outside world through the keyboard, teletype, light pen, and communication line.

We began the investigation with a study of the feasibility of communicating at the design goal rate of 40,000 bits/sec. The hardware of both the IMLAC and the Univac 1108 were capable of operating at this speed, but it was not clear that the IMLAC software could handle data coming in at the projected rate. The IMLAC minicomputer has a very limited set of operations; for example, in a single instruction, the accumulator can be shifted only one, two or three bit positions. We identified various design questions which would have to be answered and resolved to answer them all in a manner which would involve the least IMLAC processing time.

We found that we could not guarantee that an overspeed condition at the IMLAC would never take place; certain programming techniques cause the display processor, which has priority in memory access over the minicomputer, to lock the minicomputer from accessing memory over extended periods of time. The communications protocol would have to be capable of detecting and correcting loss of data due to overspeed, as the hardware has no means for detecting loss of data. Furthermore, communication links are subject to occasional error and failure, and loss of data on account of these error conditions must also be corrected.

There are two general techniques for error detection and correction in communications links: the message in error may be re-sent, or enough redundancy may be supplied in the message to allow the errors to be corrected after receipt. The second of these alternatives was rejected because error-correcting codes involve a loss of effective transmission speed due to redundancy, and they require a great deal of processing of the message as it is received. We expected a very low error rate so that the alternative of re-sending messages found to be in error is plausible, and it has the advantage of requiring only minimal processing of the received message.

One typical data transmission technique uses vertical and horizontal parity bits to detect loss of data. The data is sent as seven bits out of eight, with the eighth bit containing the parity of the other seven. We expected that loss of data due to overspeed would occur much more frequently than loss of one or more bits through a transmission error. In an overspeed condition, the vertical parity bit appended as the eighth bit of each group of seven data bits serves no useful function because the entire group of eight bits is lost.

Furthermore, the IMLAC memory is composed of sixteen bit words, and three groups of seven data bits would be required to completely fill a memory word, with five bits of the final group being ignored. The effective transmission rate would then be 67% of the actual transmission rate. The other alternative, of sending 16 groups of seven bits which would exactly fill seven sixteen bit words, was rejected as requiring too much processor time in the minicomputer, especially in view of the limited range of shift instructions noted previously.

Therefore, it was decided that the data would be sent as an eight out of eight bit code, requiring two groups of data bits to fill each IMLAC word. Error checking would be accomplished by using a function which computed a check-

sum for the entire message and appended this checksum to the end of the message. This led to another problem, which is that most communication protocols make use of special codes which signal the beginning and end of messages; in particular, one special code, a synchronization character, is necessary for the hardware to correctly reassemble the serial bit stream into groups of eight bits. If all eight bits of the code are used for data, then these special characters can be found in the message on occasion, and this must not disrupt the communications protocol.

There are currently two proposed standards for communication protocols which use an eight out of eight bit code: the DDCMP protocol developed by the Digital Equipment Corporation [3] and the SDLC protocol proposed by IBM [4]. The SDLC protocol was rejected because it would have required the construction of a special interface which eliminates certain sequences of bits by insertion of a single 1-bit as the message is transmitted and deleting these inserted bits as the message is received. The DDCMP protocol could be implemented without hardware changes to either the Univac 1108 or the IMLAC.

In the DDCMP protocol, messages are of two types: protocol and data. Each data message is divided into two parts: a fixed length header which contains the length of the second part of the message and other error detection and synchronization information, and the body of the message which is variable in length and which follows the header. Both parts of the data message have error check codes appended to them, so that the length can be checked for error prior to receipt of the body of the message. And, since the length of the body of the message is known before the message body is received, there is no need for a special end-of-message character.

Two fields in the header of a data message indicate the sequence number of the data message and the sequence number of the last message correctly received by the sending station. Loss of an entire message can be detected by finding a message which has a sequence number two or more greater than the number of the last message correctly received. The second sequence number field tells the receiving station what messages have so far been correctly received by the sending station and thus eliminates the need for a special message which acknowledges the receipt of each message (the acknowledge, or ACK, message).

The main use of the protocol message, which has a fixed length that is the same as the data message header, is to notify the sending station that a message was received which is in error and to cause the sending station to re-send the data message. If station A sends a data message with sequence number 15 to station B, and B does not receive the data message correctly (i.e. the computed check data is not the same as the check data appended to the message), then B sends a negative acknowledge (NAK) protocol message to A which contains the sequence number 14. Station A, upon receiving the NAK message from B, immediately re-sends all messages having a sequence number greater than 14. Other protocol messages correct sequence number synchronization errors and allow for initial startup of the communications link.

The DDCMP protocol has two disadvantages which we felt should be corrected for our application: one, it is designed to be used with input and output

buffering techniques and although the Univac would have no difficulty in managing input and output buffers, we doubted that the IMLAC would have sufficient processing power for this task; and, two, the DDCMP protocol uses a CRC-16 polynomial to generate the error check code. The CRC-16 check polynomial computation involves a shift and an addition operation for each bit of the data message, and it was clear to us that the IMLAC would not be able to compute this polynomial at the target data transmission rate.

The first disadvantage was overcome by making the Univac 1108 responsible for buffer management in the IMLAC; this implies that the Univac must know where each message is to be stored in the IMLAC memory and that it must specify the IMLAC memory address for each message sent to the IMLAC as a part of the message. The memory address was incorporated in the header part of data messages by extending the size of the header by two bytes. The check code for the header thus verifies the correctness of both the message length and the message address. The IMLAC message processor software is given the message length and the memory address at which the message should be placed before the body of the message is received; as the body of the message is received it is placed directly into the predetermined locations in the memory. A data message coming from the IMLAC to the Univac 1108 contains the starting address of the message body in the IMLAC memory, so that the Univac can be aware of where the message originated in the memory of the IMLAC.

We also decided on a sixteen bit sum function with end-around carry as the appropriate choice for an error check code generating function. This function is reasonably simple to compute and is secure enough for our environment.

## IMPLEMENTATION

Having made these design decisions, the software for the IMLAC was designed for our modified version of the DDCMP protocol. Some timing computations were made assuming that the display processor would be running continuously in increment mode; in this mode, it steals every other memory cycle and slows the minicomputer to half its normal operating speed. At the target data rate of 40,000 bits per second, the IMLAC would be being interrupted 5,000 times per second to process eight bit data groups, giving 200 microseconds as the maximum time to process each data byte. The effective rate for memory cycles for the minicomputer is one cycle every 3.6 microseconds, allowing a maximum of 55 memory cycles for the processing of data interrupts for both input and output.

We minimized the processing time associated with each data byte by designing the input and output processes as finite state machines which automatically sequence from each state to the next state. This design was especially easy to implement on the IMLAC because of certain memory locations which are automatically incremented each time they are referenced. These indexing memory locations address a table of jump instructions which allow the proper processing routine to be entered in only three memory cycles.

The interrupt processing routine places interrupts from the communications interface as the highest priority, with light pen, keyboard, asynchronous

communications interface, and display refresh interrupts at a lower priority level. After carefully coding each process for message receipt and transmission, we were able to verify that messages could be sent at full speed over both lines of the communication link without overspeed. Less than ten percent of the processor is available for handling other interrupt conditions when this is occurring, so the display may flicker slightly under these worst case conditions.

The format for the revised DDCMP protocol is shown in the appendix and Figures 2 and 3 show the state diagrams for the input and output processes respectively.

Having established the feasibility of communication at the target rate, we investigated various means of connecting the IMLAC to the Univac. Modems which can send data at 40,000 bits per second are quite expensive, so we decided to hard wire the IMLAC to the Univac. One of us (Nelson) designed and built a clock source and we acquired 200 feet of special, low capacitance, computer grade cable. Anticipating that some applications might be able to tolerate a lower data rate on the IMLAC to Univac side of the communications link in return for additional interrupt processing power in the IMLAC minicomputer, the clock source was made so that the transmission rates could be set by means of switches which give a range of speeds from 75 bits per second to 40,000 bits per second. After some difficulties with the Univac hardware, the hardware connection was made operational in February, 1975.

Since that time, we have run several tests of the communication link and the IMLAC software; in one test, a block of known data was circulated between the Univac and the IMLAC for over an hour at 40,000 bits per second without error; so we feel that the hardware connection is quite stable and error free.

A realtime communications handler was developed for the Univac 1108 so that it could send and receive messages over the communications link to the IMLAC. Under EXEC-8, communications handlers must be realtime programs and they are never swapped from memory. It is not feasible to have a large program locked into memory, as this would very seriously degrade the performance of the system. The 1108 software was designed so that only the handler program, which is about 4K words, needs to be locked into memory; the user program which creates display files runs as a regular conversational program and may be swapped.

EXEC-8 did not provide a mechanism which would allow the communication handler to pass messages to the user program and vice versa. Only an extremely small number of operating systems provide such a mechanism to the general user. We determined that two approaches to implementing a message passing facility were possible: an executive routine could be written to allow one program to send another program a message, or the two programs could share a common area of memory. The former method was felt to be less desirable than the latter because it would require implementation of a new executive function, with the attendant system maintenance difficulties, and because it would have required a substantial enlargement of the resident executive part of EXEC-8.

The implementation of the message passing facility proved to be an interesting exercise in design. Message traffic in one direction can be visualized as a



two stage producer/consumer problem. One program places a message in the common area and then notifies the other program via a semaphore that a message is available. The other program becomes eligible for processor time and eventually removes the message from the common area. Since messages are passed in both directions simultaneously, four activities are required to transfer messages.

If one thinks of each activity as a message pump, then it is immediately apparent that each activity has the same function as the other activities, differing only in the location from which it obtains its input and to which it sends its output. Figure 4 illustrates this. With this in mind, we designed the pump program as a general, re-entrant routine, and it is located in the common area where it is executed by all four activities simultaneously.

The software and communications link has been functioning for over three months now, and we have been able to successfully demonstrate simple animations of stick figures. The animations are created by the 1108 and demonstrate the ability of the 1108 to rapidly alter a display program running in the IMLAC via the high speed communications link.

All of the difficulties which we have encountered thus far have been with the Univac software and hardware. In the message passing facility, we have uncovered what is apparently a design flaw in EXEC-8 that causes occasional system crashes, and we expect that the Univac personnel who have been helping us will have this problem fixed shortly.

The next stage in the development of the graphics subsystem will result in an increase in hardware sophistication and in the implementation of a display file compiler that is designed for implantation in the FORTRAN and COBOL libraries, and in special versions of the LISP, APL, and BASIC interpreters. The hardware development will place the communications protocol handler in an outboard microprocessor which will transfer data to and from the IMLAC memory directly, thus freeing the minicomputer in the IMLAC of the burden of communications processing and facilitating the response of the IMLAC to the light pen and other devices (Figure 5).

#### ACKNOWLEDGEMENTS

The authors wish to acknowledge the support of this project by The Energy Institute of The University of Houston, and by the Computing Center of the University of Houston. The development of further software and applications is being supported by the National Science Foundation, Grant number, DCR 74-17282.

## REFERENCES

1. Newman, William M., and Sproull, Robert F., Principles of Interactive Computer Graphics. McGraw-Hill, 1973.
2. IMLAC Corporation "IMLAC PDS-1D Programming Guide," 1973.
3. Wecker, Stuart "A Message-Oriented Protocol for Interprocessor Communication," Digital Equipment Corporation, May 1974.
4. IBM Corporation "IBM Synchronous Data Link Control General Information," Form Number GA27-3093-0, March 1974.



## APPENDIX

### PROTOCOL MESSAGE DESCRIPTIONS

#### Data Message Description and Format

The elements in a data transmission are as follows:

SOM, device address, flags and count, response, message number, memory address, check 1, data, check 2

Note: Preceding the SOM character, sync words are transmitted (octal word 026). These characters cause the receiving port hardware to synchronize to the incoming data so that each successive 8-bit word will be correctly received.

SOM	- the Start Of Message character (8-bits). This, or DLE, must be the first non-sync character to insure that the receiving port has not started due to a false sync word.
device address	- the 8-bit quantity used to specify which device a message is for in a multi-device system
flags and count	- the 16-bit quantity containing a 13-bit count of the number of 16-bit words in data and three bits used as control flags, if required
response	- the response from a device giving the number of the last received error-free message
message number	- the number (in modulo 256) which is assigned consecutively to each transmitted message
memory address	- the 16-bit address in PDS-1 memory at which the received message storage is to begin or from which a transmitted message reading began
check 1	- the 16-bit checksum of the message header formed by adding each successive 16 bits with end-around carry
data	- the 16-bit data elements, the number of which was specified in flags and count
check 2	- the 16-bit checksum of the data elements

#### Non-data or Protocol Message Descriptions and Formats

Protocol messages are header-only messages which are used for system control and error recovery. The three types of messages are: (1) message acknowledgement or ACK, (2) erroneous message acknowledgement or NAK, and (3) a general purpose message or MES.

The elements of an ACK message are as follows:

DLE - the Data Link Escape character. This indicates that a header-only message follows.

ACK type - identification of an ACK message

fill 1 - 8-bit fill word  
fill 2 - 16-bit fill word  
check - message check word

The elements of a NAK message are as follows:

DLE, device address, NAK type, error, response, message number, fill 2  
check

NAK type - identification of NAK message  
error - the type of error found, if required

The elements of a MES message are as follows:

DLE, device address, MES type, message, response, message number, fill 2  
check

MES type - identification of MES message  
message - 8-bit field conveying the required message

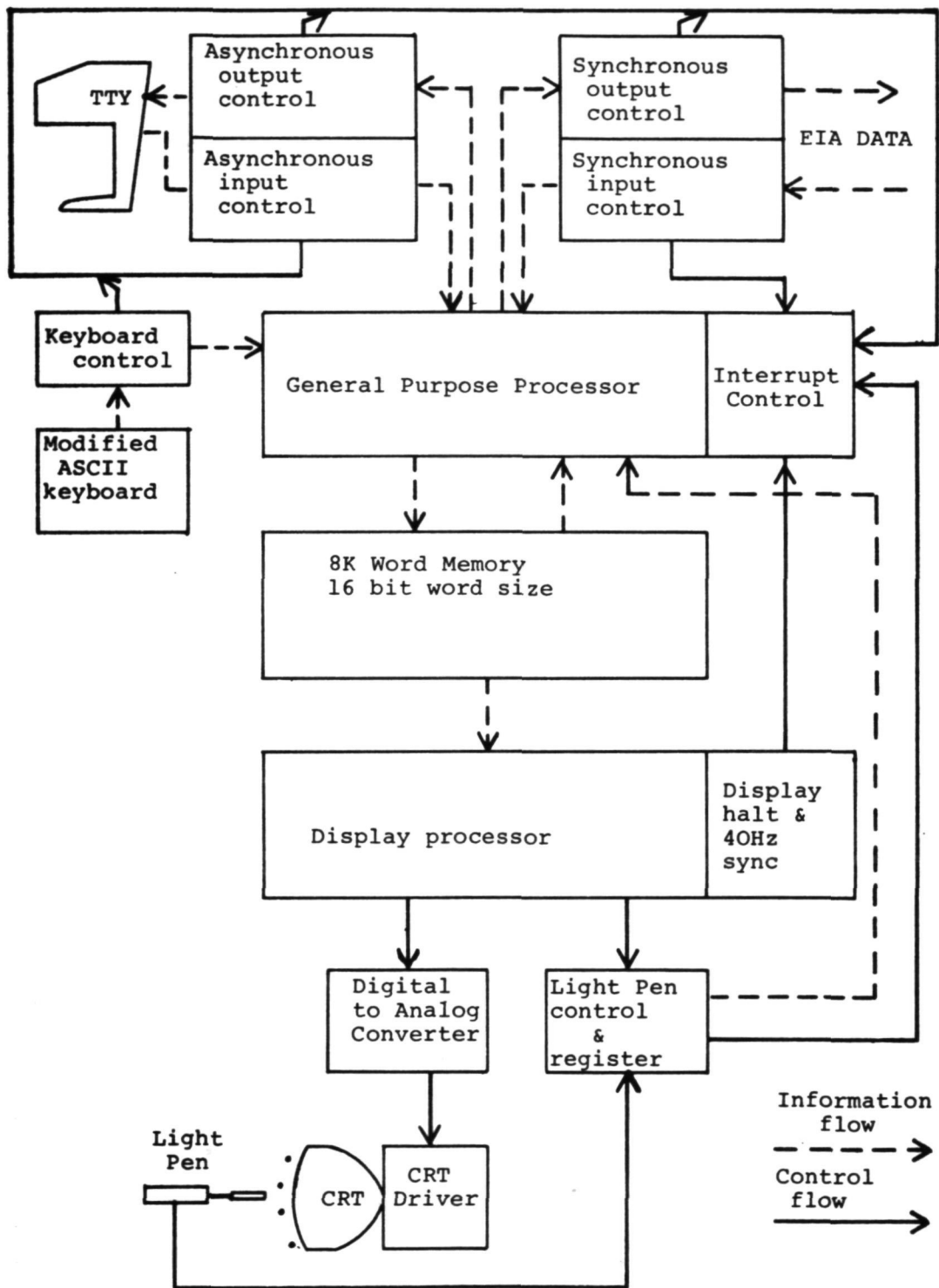


Figure 1.- IMLAC PDS-1D organization.

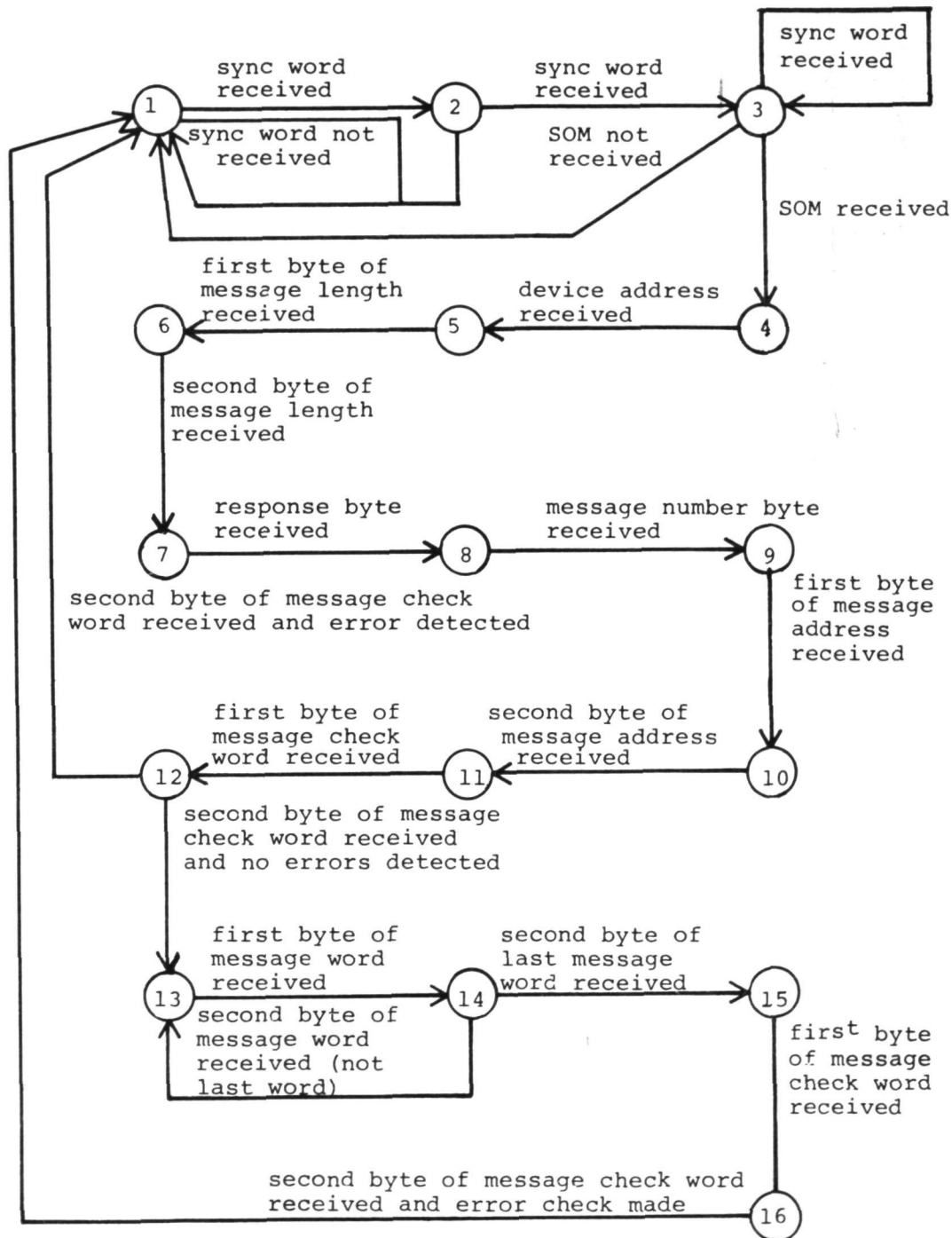


Figure 2.- Receive state transition diagram.

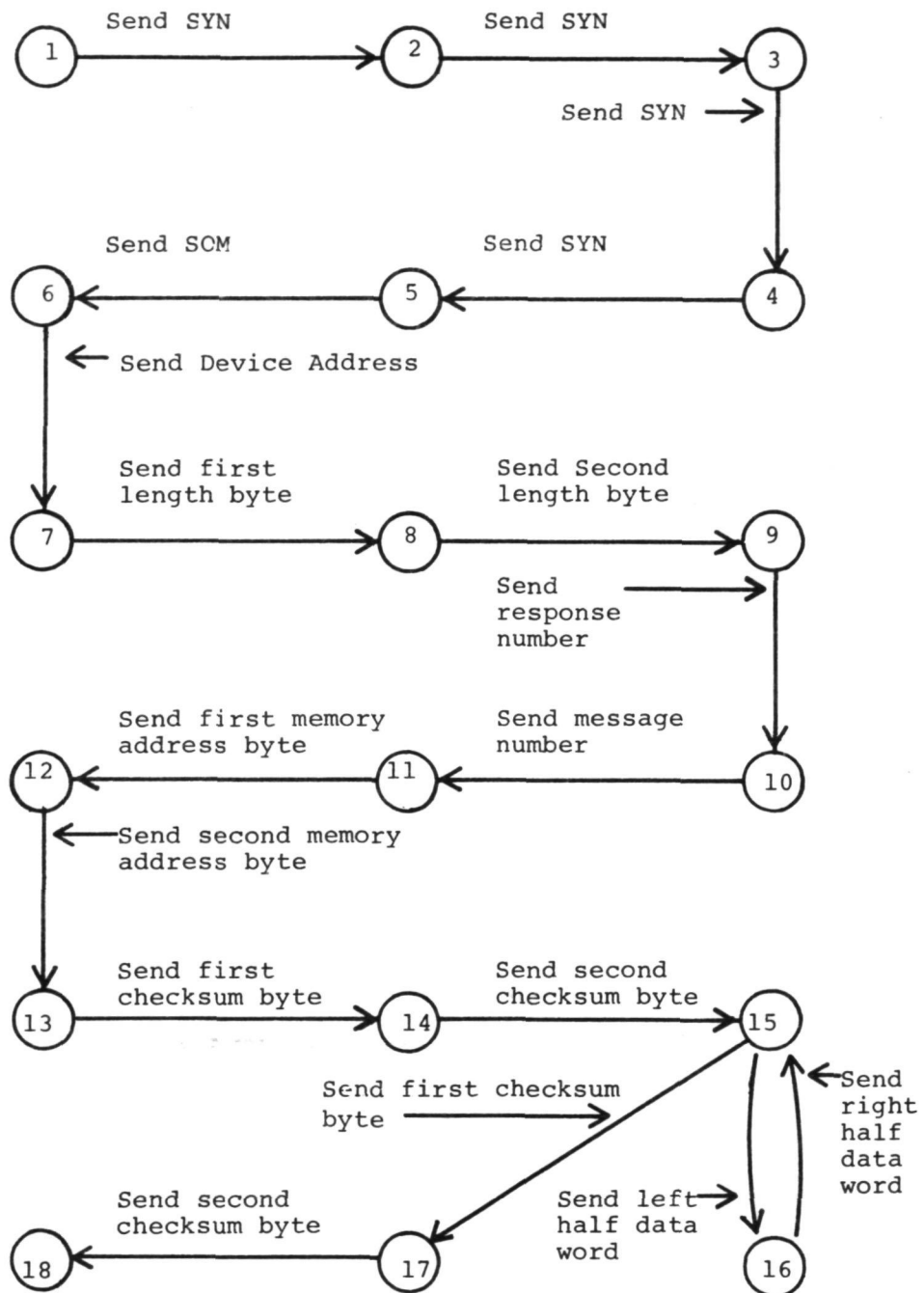


Figure 3.- Send state transition diagram.

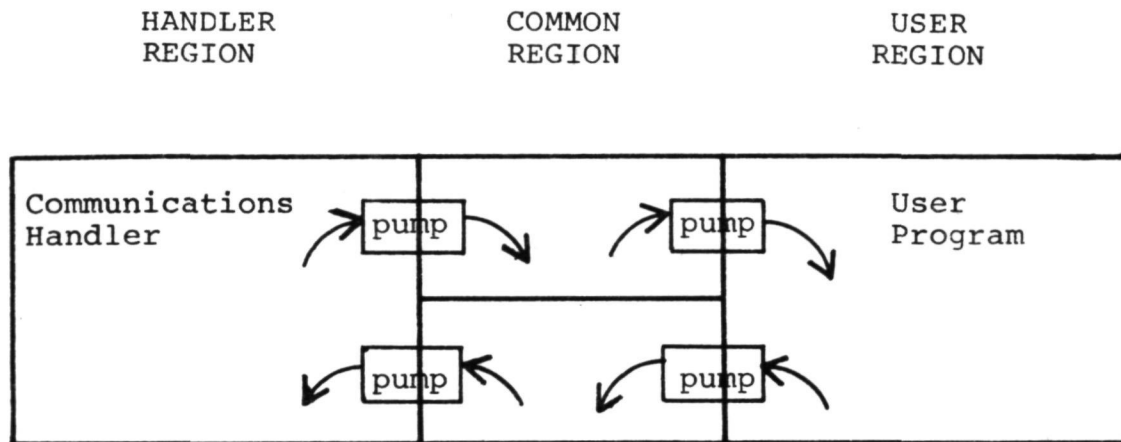


Figure 4.- Interprocess communication scheme.

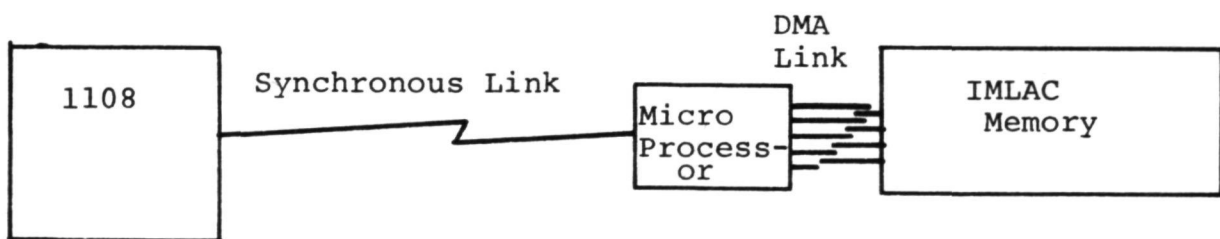


Figure 5.- Proposed data link hardware.

Page Intentionally Left Blank

## BIG SYSTEM - INTERACTIVE GRAPHICS FOR THE ENGINEER

Charles E. Quenneville

Boeing Computer Services, Inc.

### SUMMARY

The BCS Interactive Graphics System (BIG System) approach to graphics is presented, along with several significant engineering applications. The BIG System precompiler, the graphics support library, and the function requirements of graphics applications are discussed. The paper concludes that graphics standardization and device independent code can be developed to assure maximum graphic terminal transferability.

### INTRODUCTION

A great many users of interactive graphics have common graphic display requirements. Some of these requirements are the abilities to display and modify input and computed program variables, to interactively control the logical program execution, to display two- and three-dimensional data in a variety of graphical forms, and to obtain offline hardcopy output. There are available in the commercial world a wide variety of graphic display terminals, each with its own unique software to support utilization of its unique terminal features. This software, though becoming increasingly sophisticated, generally provides only the basic capabilities for displaying random characters and vectors. This requires that the application programmer become familiar with the terminal software in order to develop software on a more convenient, though more costly, level. Further, once the application has been developed, it becomes "locked" into a particular terminal and its portability to other graphic terminals is seriously limited. It is becoming increasingly important that we identify functional requirements between graphic programs and graphic terminals, and provide standard graphics software which can automatically perform these functions.

The BIG (BCS Interactive Graphics) System (ref. 1) is such an approach to standardization in the field of interactive graphics software. BIGS at Boeing dates back to 1967 when graphics first became available to the engineering community as an application tool. After a short period of application development, it was readily apparent that the development of graphics within an application was by no means a trivial task. We identified four major functional requirements that were common to most of our graphic users: (1) logical control over program execution; (2) high level graphic display



capabilities; (3) ability to display and modify the value of program variables and arrays; and (4) offline hardcopy output. The ability to draw arbitrary lines and characters did not meet all the requirements of our graphics applications. In the fall of 1967 Boeing began the development of a standardized graphics system to meet these functional requirements through a project known as CADLIB (Computer Aid to Design Library). A precompiler was also introduced to help eliminate the tedious task of generating the data statements required when creating displays with textual information. Over the years the name CADLIB was changed to BIG System and the system was improved and expanded until it is today accepted and actively used by Boeing, BCS and commercial clients.

The BIG System is presently operational on the IBM 360/370 with the IBM 2250 graphic display terminal, the CDC 6600 with the CDC 241/243 graphics terminal, the Digital Equipment Corp. PDP 11/45 with either the Vector General 3-D hardware four color system or the Tektronix 4010 series display terminals, and the Prime 300 with Tektronix display terminals. The BIG System is also fully operational on BCS's CDC and IBM timesharing networks (MAINSTREAM-EKS and MAINSTREAM-CTS) using Tektronix 4010 series terminals.

#### THE BIG SYSTEM PRECOMPILER

The BIG System precompiler translates macro-level, graphics oriented statements into FORTRAN code which can then be processed by the FORTRAN compiler and linked to the graphics library to form executable code (see Figure 1). The precompiler is operational on both the IBM 360/370 and the CDC 6600 series computers, and it provides complete transferability of the graphics portion of programs written on those machines. The precompiler provides the facility by which an unsophisticated computer user can easily add graphics to his programs. It is not necessary that he know all the details of the graphics system, simply that a given macro-statement will produce a given type of display. It is an important distinction that a system implementor is more interested in how something is done, while the user is interested in what it does.

The precompiler language consists of macro-level display statements which provide the capability for controlling the logical execution of the program and displaying and/or modifying program variables.

One such powerful macro-level display statement for controlling the execution flow of the program is the "DISPLAY MENU" statement. A "menu" can be generated which consists of a number of lines of text, each one of which represents an option in the program. By selecting a line in the menu, control is transferred to a corresponding point in the program. Other devices for directing program control will be discussed later in this paper.

The use of the precompiler for display and program control are best illustrated with an example:

The equation for a projectile fired from a gun at an initial angle of inclination (A) with a fixed muzzle velocity (V) over an interval (T) under a constant acceleration (g) is given in the following equation:

$$X = VT \cdot \cos(A)$$
$$Y = VT \cdot \sin(A) - .5gT^2$$

The program listed below allows for the display and modification of the values of V, A, and plotting step size; the plotting of the values of X vs Y for a range of T's; the display of the X and Y arrays so they may be modified; and the program control over data display and program execution.

```
DIMENSION X(50), Y(50)
DATA NPTS/50/
DATA V,A,TINC,G/600.,30.,.3,32.2/
START DISPLAY
1  DISPLAY MENU $PROJECTILE$
   1  :DISPLAY CONTROL PARAMETERS: (10):SOLVE:(20)
   2  :DISPLAY TABLE: (30)
      GO TO 1
10  DISPLAY DATA *VELOCITY*V,*ANGLE IN DEGREES*A,TINC
      GO TO 1
20  T = 0
      DO 21 I=1,NPTS
         X(I)=V*T*COS(A/57.3)
         Y(I)=V*T*SIN(A/57.3)-.5*G*T**2
21  T = T + TINC
      DISPLAY PLOT LINXLINY $PROJECTILE PLOT$ X,Y,NPTS
      GO TO 1
30  DISPLAY TABLE $PROJECTILE TABLE$ X,Y,NPTS
      GO TO 1
      END
```

The resulting displays are shown in Figures 2 through 5. Interactive commands are automatically provided for control, variable update and offline plotting. The "START DISPLAY" command performs all the initialization required by the graphics terminal. Statement Number 1 and the corresponding Figure 2 illustrate the "DISPLAY MENU" statement. The simple format provides for titles enclosed in dollar signs (\$), and lines of text enclosed in colons (:) and statement numbers enclosed in parenthesis [()]. When a select is made on a particular line, program control is automatically transferred to the associated statement number.

Statement Number 10 in the example and the corresponding Figure 3 illustrate the simplicity with which program variables can be displayed. The program variable names are listed separated by commas (,) following DISPLAY DATA. The display is automatically generated with the variable name followed by its value. The variables can be modified at the terminal. When the user exits from the display the variables are modified within the program. Variable names may be replaced by up to twenty characters of text enclosed in asterisks (\*) preceeding the variable name.

The "DISPLAY PLOT" statement automatically scales the data and generates the plot shown in Figure 4. The grid is drawn and "good" grid values are automatically generated for maximum resolution. This reduces the time to program the operation and reduces the probability of errors.

The DISPLAY TABLE statement (see Figure 5) illustrates another type of data display statement. A table of X and Y values is displayed; although only the first twenty points are displayed, the user may reposition this twenty point window to view any of the remaining points. He may also plot and/or modify the data points. This macro-statement in effect invokes a mini-editor that allows the user to manipulate and replot the data. The data is modified only in core. It is the user's responsibility to update any disk or multiple files.

#### THE GRAPHICS DISPLAY LIBRARY

Each of the hardware, software or user directed operations are implemented by one or more calls to library subprograms. These subprograms provide capabilities for accurate, fast and easy generation of the display portion of the application program. In order to provide for the device independence described above, the system consists of two types of subprograms: those that communicate with the terminal (called the low-level routines) and those that manipulate the user's data (called high-level routines). A simple, flexible interface between the two levels allows the implementor to exchange low-level routines when he changes terminal systems. The low-level routines are provided by the terminal vender (ref. 2) and simply draw lines, erase lines, and change the terminal state.

The higher level routines are the routines called by either the user or the precompiler in order to express the sequence of operations required to generate an image display or a user interaction. These allow the user to select and define defaults for a graphics terminal (initialization), and to define the logical program control and the display format. The program control can be provided through menus, function keys, input options and other input devices such as keyboards, light pens, data tablets and crosshairs. On terminals not equipped with certain devices, the control functions can be simulated. For example, a function key interrupt on key number 13 can be simulated by inputting "KEY, 13" on the keyboard of a terminal not equipped with a function keyboard.

Graphic displays are generated by defining a coordinate system (rectangular or polar), scales (linear, logarithmic, etc.), labeling and text generation. The scales are determined either automatically in order to fit all the data or a section of the data (blowup), or explicitly by the user. Where data falls outside the screen scale, a scissoring routine is used to exclude all elements outside the region. Where the data are three-dimensional, routines are available to rotate the objects. If the terminal is equipped with 3-D hardware, the display is generated directly with rotation provided automatically through dials, a joy stick or other input devices. On terminals not equipped with three-dimensional hardware, an orthographic projection is implicitly performed to project the data into a two-dimensional coordinate system. Textual data display and editing routines are also provided for displaying and modifying program variables, data arrays, tables and matrices.

Since all terminals do not have a device to generate a permanent copy of the displays generated, a technique to preserve a representation of the object on the screen, for later hardcopy plotting, has been developed. Interactive plot commands are available to the user to selectively create plots on the Stromberg-Carlson (SC4020), the CalComp, Gerber, or other remote plot devices, by means of a post-interactive processor.

#### BIGS USERS

Within the Boeing Company the BIG System has been used in many application areas such as: Finite Element Structural Analysis, Antenna and Radar Design, Control Systems, Weapon Systems, Flight Simulation, Rapid Transit, Project Management, Master Dimensions, and Data Analysis and Reduction. Two of these significant applications are the STRIP and GGP programs.

## STRIP

STRIP (Structural Interactive Plotters) is a graphically oriented interactive preprocessor for NASTRAN (NASA Structural Analysis) used for verifying the basic geometry of a NASTRAN defined structural model (grid locations and element connections). A similar postprocessor for NASTRAN has also been developed for displaying the NASTRAN output (refs. 3 and 4).

The data used to define a NASTRAN model and the results generated by a NASTRAN analysis are inherently graphical. This is one of the reasons the plotting capability provided within NASTRAN has been so widely used. The major problem with that plotting package is one of computer operations. The time required to get a NASTRAN job scheduled, run on the computer and the plots returned is often several days. This is too long a time for an analyst to effectively check his input. As a result many expensive NASTRAN runs are made that solve the wrong model.

STRIP was developed using the BIG System and operates on BCS's CDC and IBM based timesharing network using comparatively low cost Tektronix terminals (see Figure 6). The program was written using both the precompiler and the BIG System library directly. The graphic system provides for all terminal initializations, graphical displays, program control and offline hardcopy output. Program control is provided through menus, simulated functions keys and crosshairs. Offline plotting (Gerber, SC4020 and/or CalComp) is provided as an automatic option.

The program logic for the NASTRAN preprocessor operates in the following way. The program reads the NASTRAN deck. The grid cards are converted to the base XYZ system and stored in data arrays. The element connectivity cards are read and an element connectivity table is built. At the time the connectivity cards are read a check is made to see that each grid reference on the connectivity card has been defined by a grid card. An error message is printed out that identifies any missing grid points. After the NASTRAN data has been read the remainder of the program is executed interactively from the terminal. This provides the user with the opportunity to select those operations that contribute most to the data verification process. Since many types of errors might occur and only the user can detect most of them this type of operation is both cost-effective (fewer spoiled runs) and productive (the user can choose to observe only the useful data displays). The user then chooses the next step from the following menu:

DEFINE DISPLAY PARAMETERS

DEFINE DISPLAY SET

DISPLAY STRUCTURE

DEFINE DISPLAY PARAMETER allows the user to choose the view angle. By defining different view angles the structure may be rotated in any direction. Detail which is hidden because of a given orientation can be interactively reoriented to show clearly the structural idealization. Often several viewing angles are necessary in order to examine all of the geometry and connectivity of the model.

DEFINE DISPLAY SET allows the user to select any part of the model to be displayed. The linear elements, triangular elements and/or quadrilateral elements within one or more ranges of element IDs may be interactively selected as the display set. The linear elements, triangular elements and/or quadrilateral elements may be selected to be displayed individually or in combination. In addition specific element IDs or a range of IDs may be selected. The set selection option allows the model to be displayed section by section. Duplicate lines can be displayed in separate views.

DISPLAY STRUCTURE causes the program to create the previously defined display with user chosen viewing angles. The user has the option to display gridpoint IDs and/or element IDs. Another important capability of this option is the BLOWUP feature. The diagonal corners of a new display window are defined as two points on the display. The window defined by these points are the new scale limits. The structure is displayed so that the window is expanded to the boundaries of the display screen. This is an important capability in viewing complex structural details.

The preprocessor described above has been extensively used. Figure 7 shows the NASTRAN demonstration problem for a 3-dimensional Delta Wing. The model includes several linear, triangular and quadrilateral elements. Figure 8 shows a NASTRAN spherical cap problem. The spherical cap model has been rotated counterclockwise so that the gridpoint IDs are more visible. Figure 9 shows a blown up portion of the same spherical cap. Notice the lines on the right and bottom are clipped at the display boundary. Planned extensions of the system include the display of single point constraints, multipoint constraints, forces and concentrated masses.

#### GGP

GGP is a general purpose data visualization program used to view, manipulate and analyze vast amounts of data. Versions of this program operate on the CDC 6600 using a 243 display terminal, a PDP 11/50 with a Vector General 3-D four color display terminal, and Tektronix terminals, and on BCS' CDC timesharing system with Tektronix terminals. A variety of application programs have been developed on the PDP 11/50; however this discussion concerns itself only with the GGP applications.

The PDP 11/50 is an independent minicomputer dedicated to the Boeing Commercial Aircraft Company's Propulsion Computational Technology Group. Its primary intent is to provide, through graphics and structured random access files, tools and techniques for engine risk assessment, fluid and structural analysis, and development of computer design systems for engine related hardware components.



The PDP 11/50 graphic display terminals consist of a Vector General (VG) display system, a Tektronix 4014 and a Tektronix 4010. The Vector General is a three-dimensional graphics display terminal with a four color monitor, keyboard, dials, joy stick and a data tablet. The installation allows the engineer to directly interact with either his data, data at the test facilities, or data at the central large-scale computer, the CDC 6600.

The full PDP 11/50 system functional configuration is given in Figure 10. The operating system software for the PDP is RSX 11D Version 4.

The BIG System provides all the graphic interfaces between the user programs, the vendor supplied software, the graphic terminals and graphic input devices. The precompiler described above is as yet not available on this installation. A user task written using BIGS may operate on any of the three graphic terminals without the necessity of rebuilding the program. The method of accessing the terminals through the BIG System is shown in Figure 11. When the user program begins execution the user has the option of selecting which graphic terminal is to be used. Once a terminal has been selected, a BIGS user task is automatically initiated to service the terminal and the user task. All graphics initialization, program control, graphic display, updating and hardcopy output are provided automatically. When three-dimensional data are generated on the VG, rotation and translations are automatically provided through dial inputs. When three-dimensional data are generated for the Tektronix, an orthographic projection into a two-dimensional display is provided automatically.

The GGP System is a system of intercommunication modules designed to permit rapid access to and analysis of vast data. The modules included within the GGP System and their functions are shown in Figure 12.

The data to be analyzed by GGP may be input in a variety of forms: cards, magnetic tape or from a RJE link to our CDC 6600 facility. The data is then sorted and rewritten into random access files. The GGP System can simultaneously display information from three similar files with each file consisting of up to 360 data sets, each containing up to 3,500 points per data set. The maximum data access capacity at any one time is 3.78 million data points.

The GGP System allows the engineer the following direct and immediate interaction with his data:

1. Immediate visualization of data in almost any desired form (i.e., linear, logarithmic or polar X-Y plots, either two- or three-dimensional).
2. Easy data editing and manipulation.
3. Easy comparison of any portion of one data set to other similar data.
4. Several curve fitting and pairing techniques.
5. Cross plotting and averaging techniques.

## 6. Hardcopy output.

Examples of these types of graphical output are shown in Figures 13 and 14.

Program control for GGP is provided primarily through the use of menus. The options available to the user at a particular point in the program are displayed in a menu. Figure 15 is an example of one of the many menus available to the user. Selections on a menu may cause either other menus or other plots to be displayed, allowing the user to control the types of plots to be displayed and the method of analysis to be performed on the data. Other menus are available that allow the user to select information for display or modification.

In addition to menus, program control is also provided through the input devices: keyboard, dials, joy stick, and data tablet. Keys are provided on the keyboards which when depressed will automatically generate a hardcopy plot.

## CONCLUDING REMARKS

Graphic display systems should have the ability to logically control the program execution, generate high level graphic displays, provide the capability to display and modify the values of program variables and arrays, and automatically create hardcopy output. Graphics standardization and device independent code should also be developed that assure machine independence and terminal transferability. Two application programs were discussed, using different host computers, operating systems, communicator networks and graphic display terminals. Though vastly different in scope and operation, the application program approaches to graphics through BIGS and the graphic routines called were, in general, identical.

## REFERENCES

1. Quenneville, C. E., BCS Interactive Graphic System, Boeing Computer Services, Inc., 10201-044, June 1975.
2. Information Display Products: Tektronix Plot-10 Terminal Control System. Document No. 062-1474-00, Beaverton, Oregon (1972).
3. Herness, E. D. and Kriloff, H. Z., NASTRAN Pre- and Postprocessors Using Low-Cost Interactive Graphics. Proceedings of the 4th NASTRANS User's Colloquium, Langley Research Centers, August 1975 (NASA TM X-3278).
4. Herness, E. D. and Tocher, J.L., Design of Pre- and Postprocessors. Proceedings of the International Symposium on Structural Mechanics Software, University of Maryland, June 1974.



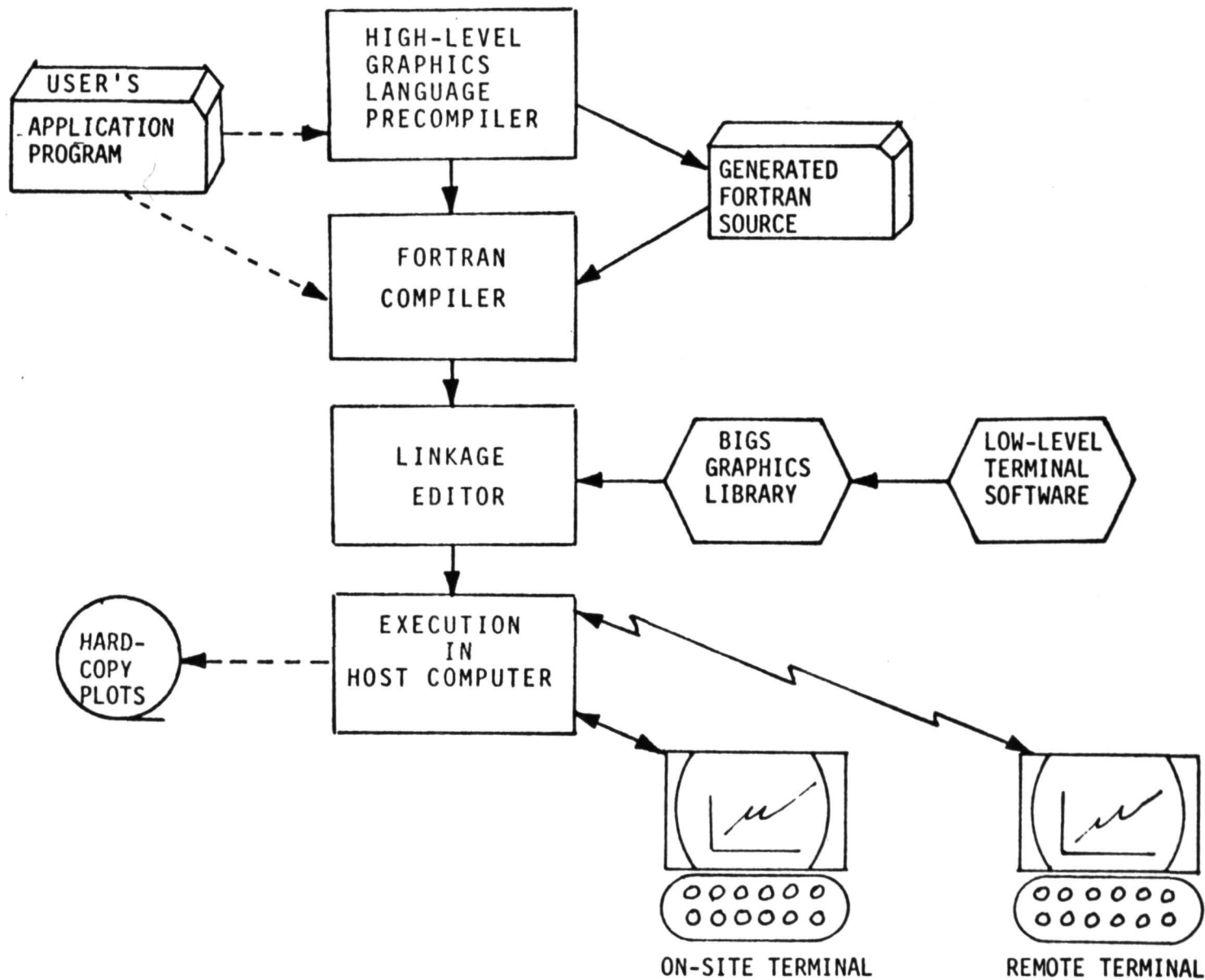


Figure 1.- BIG System Configuration - Large Scale Computers.

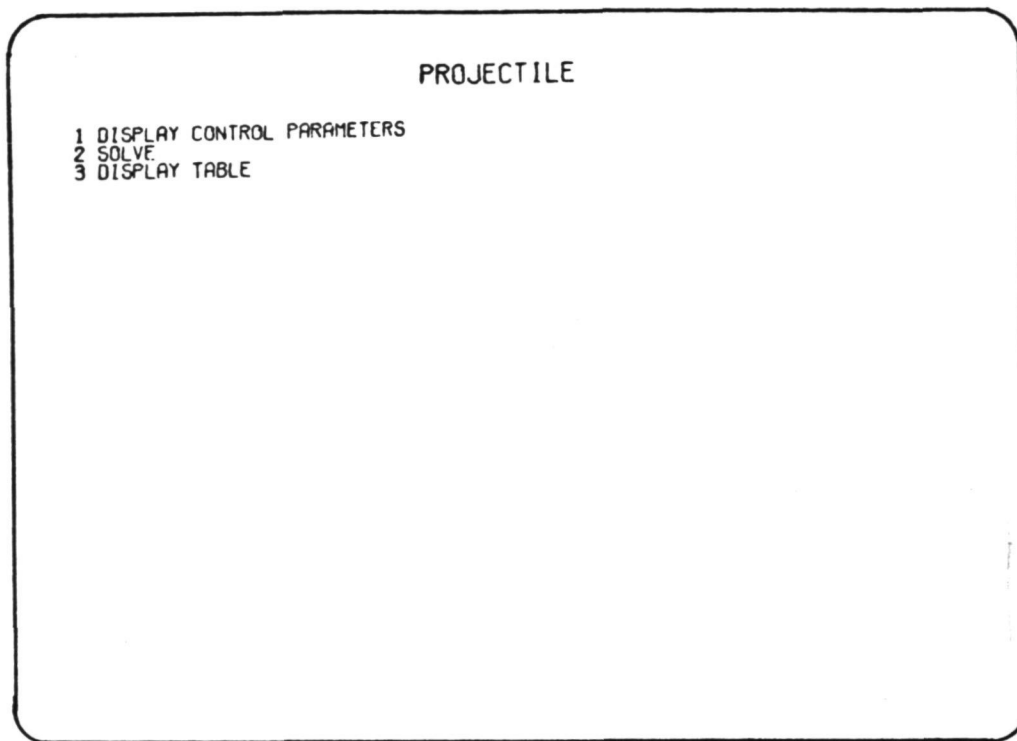


Figure 2.- Example of Display Menu.

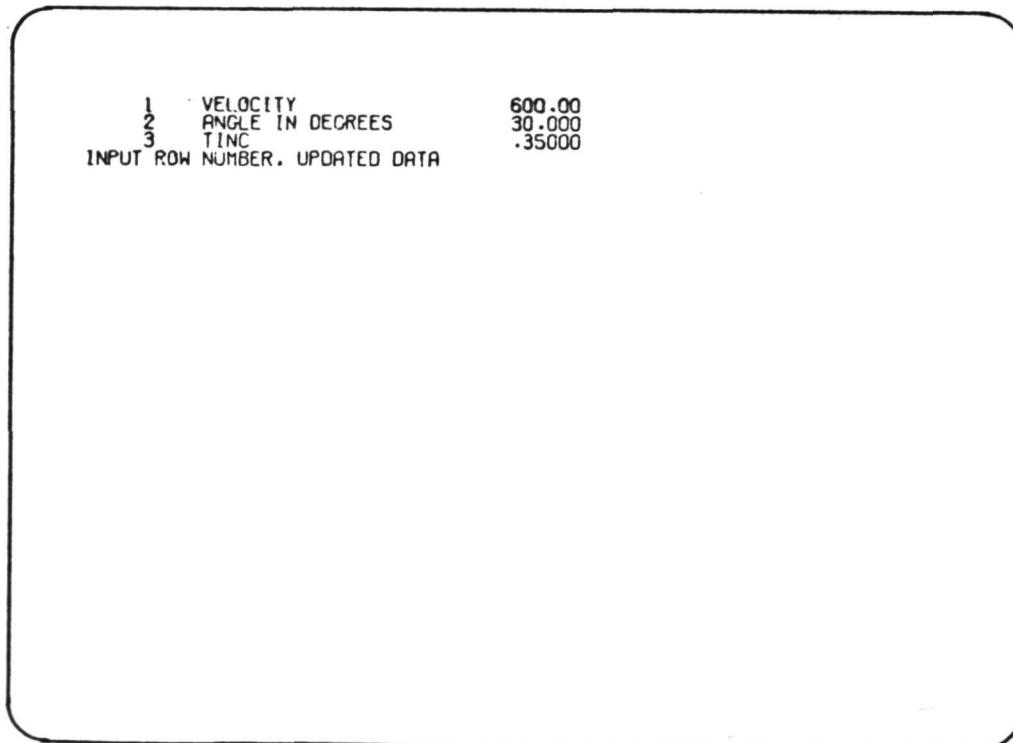


Figure 3.- Example of Display Data Statement.

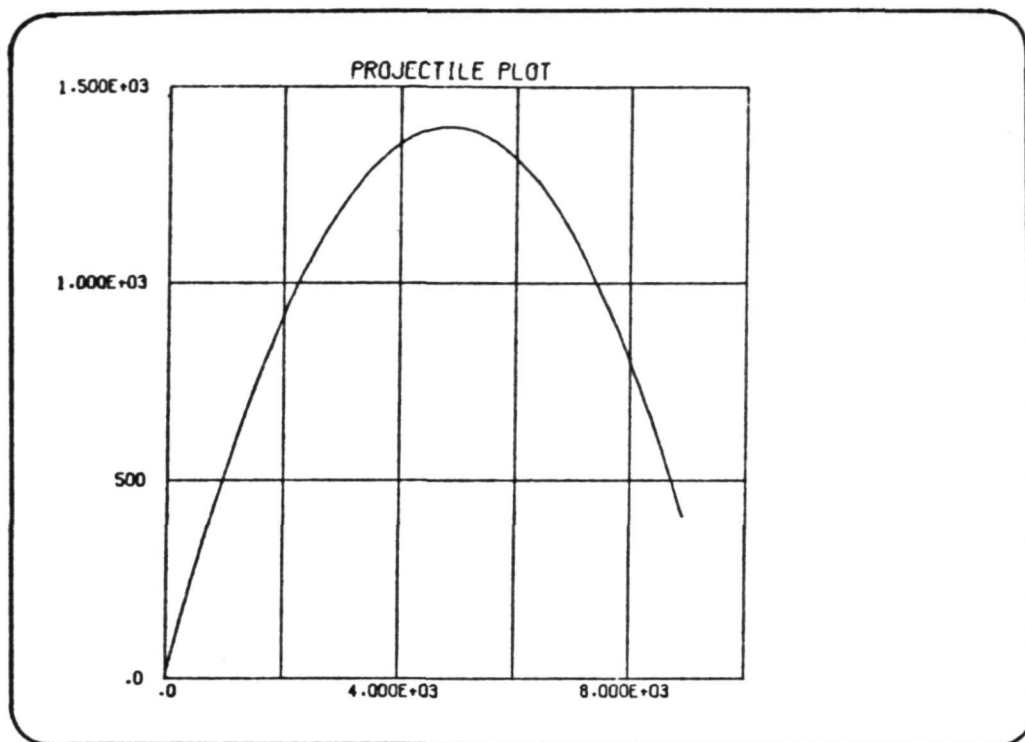


Figure 4.- Example of Data Plot.

PROJECTILE TABLE

NUMBER OF POINTS INDEPENDENT	SO DEPENDENT
1	.0
2	181.87
3	353.74
4	545.61
5	727.48
6	909.35
7	1091.2
8	1273.1
9	1455.0
10	1636.8
11	1818.7
12	2000.6
13	2182.4
14	2364.3
15	2546.2
16	2728.0
17	2909.9
18	3091.8
19	3273.6
20	3455.5

OPTIONS: REPLACE,I,X,Y DELETE,I INSERT,I,X,Y UP,N DOWN,N UPDATE  
LN LN LG LN LG LG PL PLG NPTS,N

Figure 5.- Example of Data Table Display.

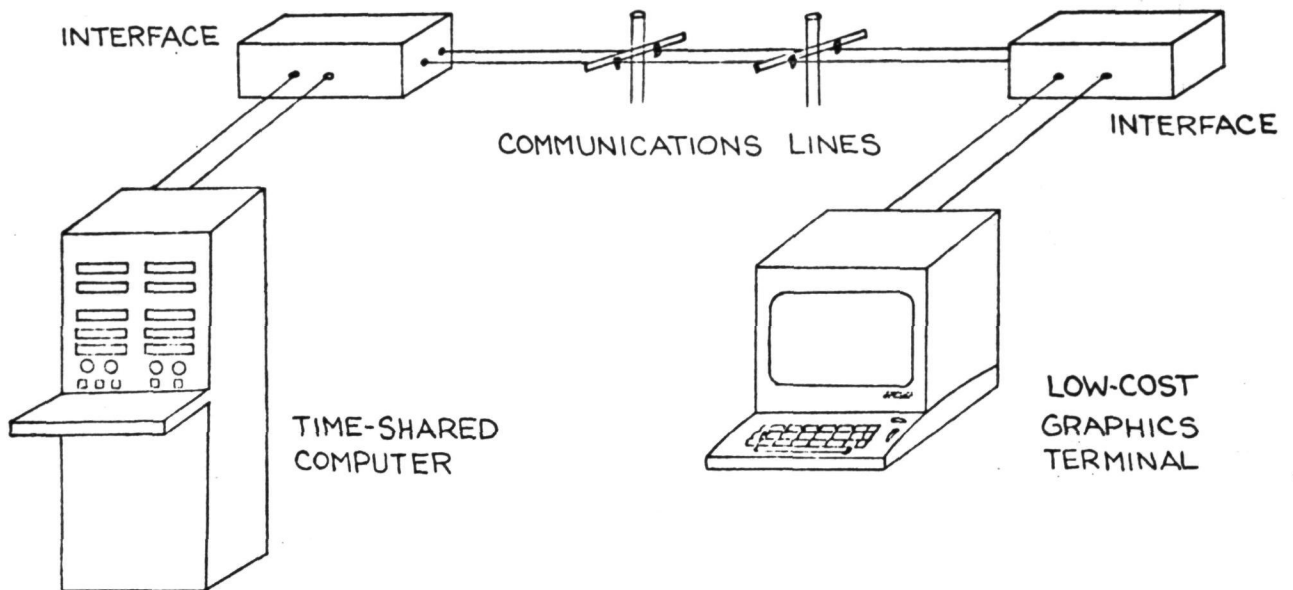


Figure 6.- Hardware Elements of a Low-Cost Graphics System.

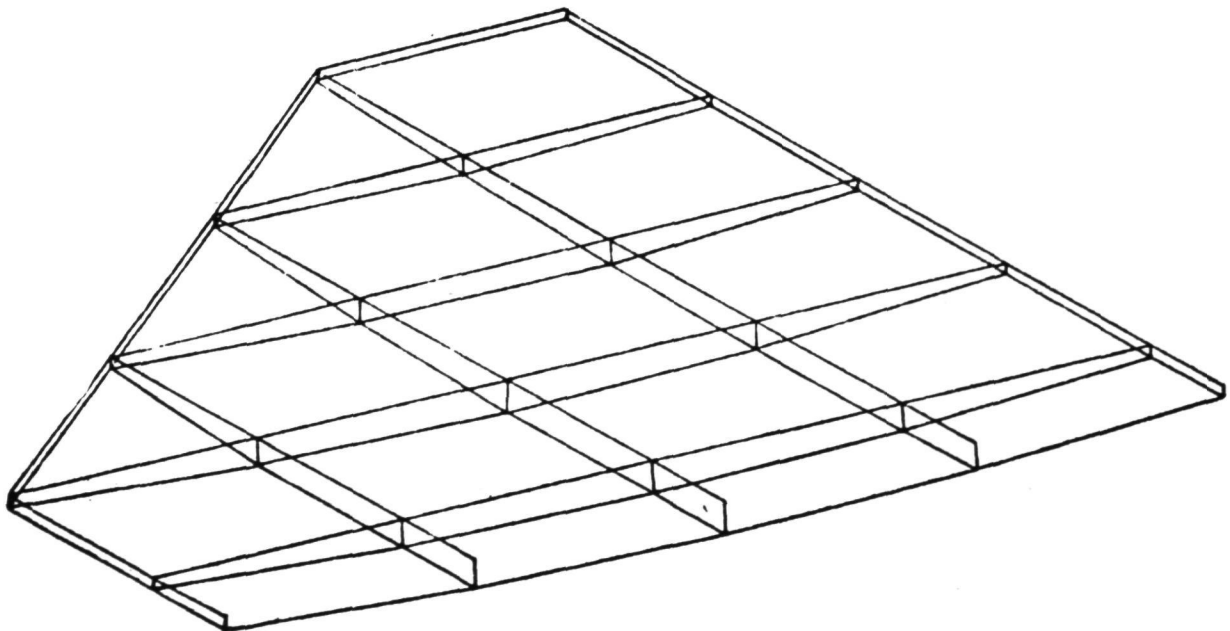


Figure 7.- Demo 1-1 Delta Wing.

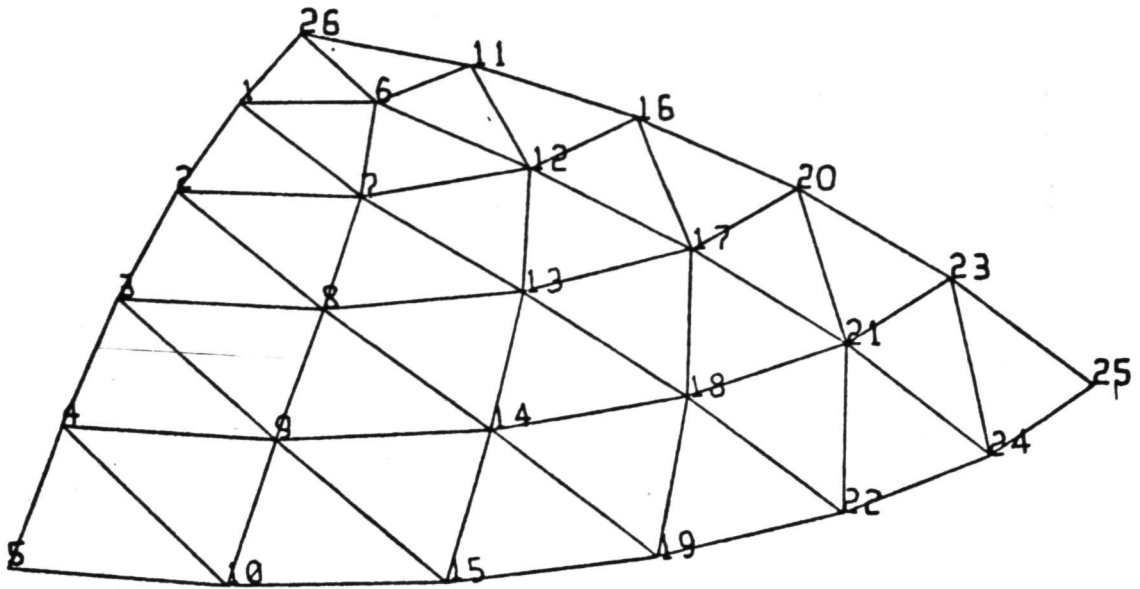


Figure 8.- Demo 1-2 with Grid ID's.

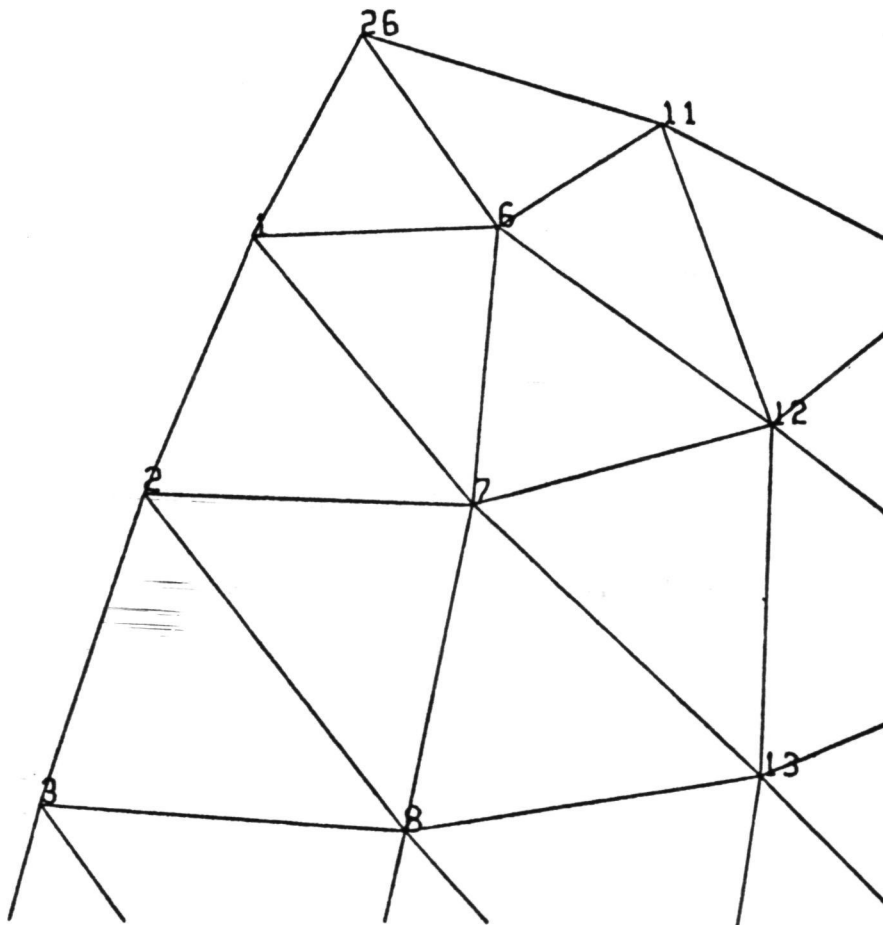


Figure 9.- Demo 1-2 Blowup.

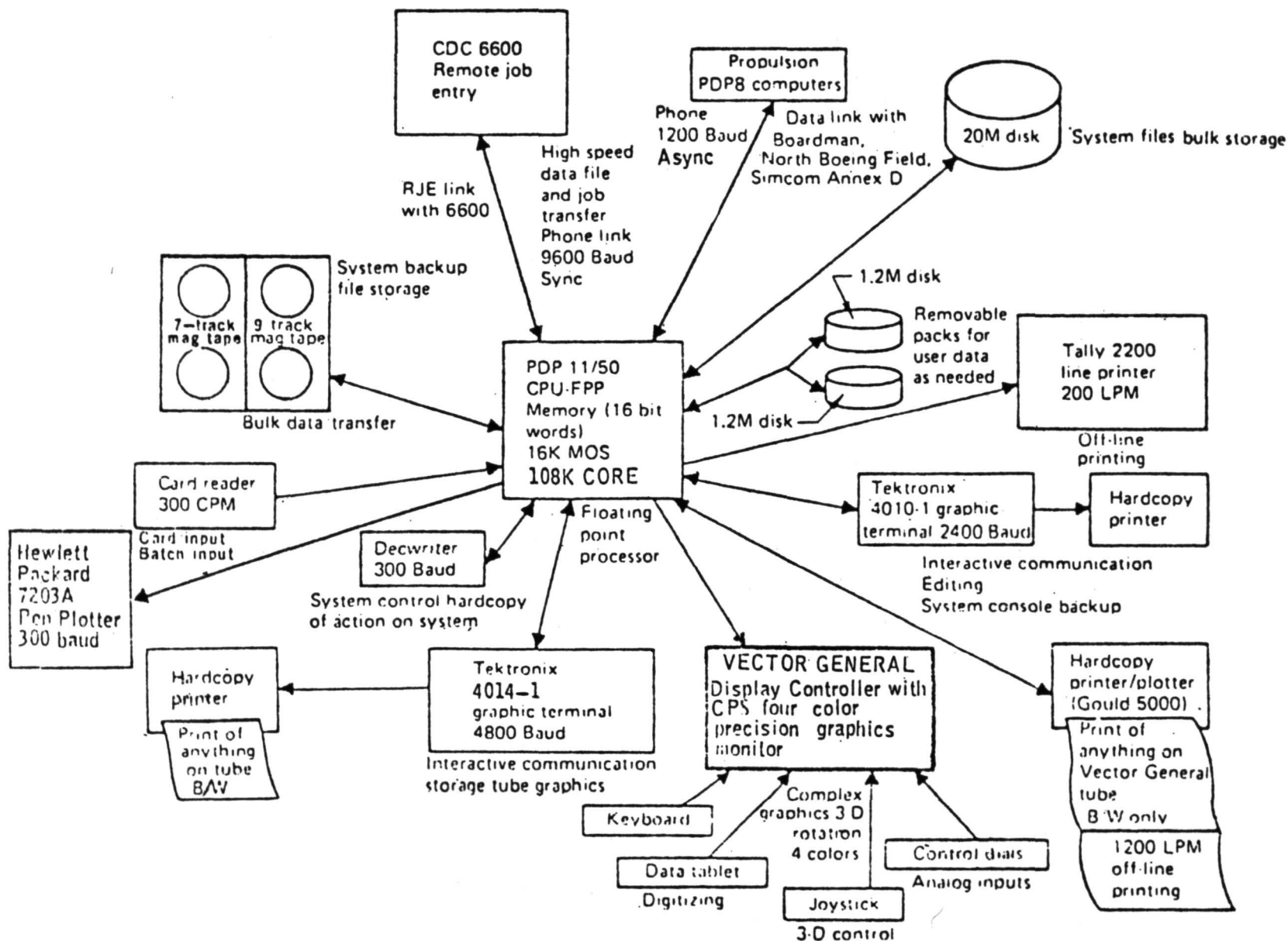
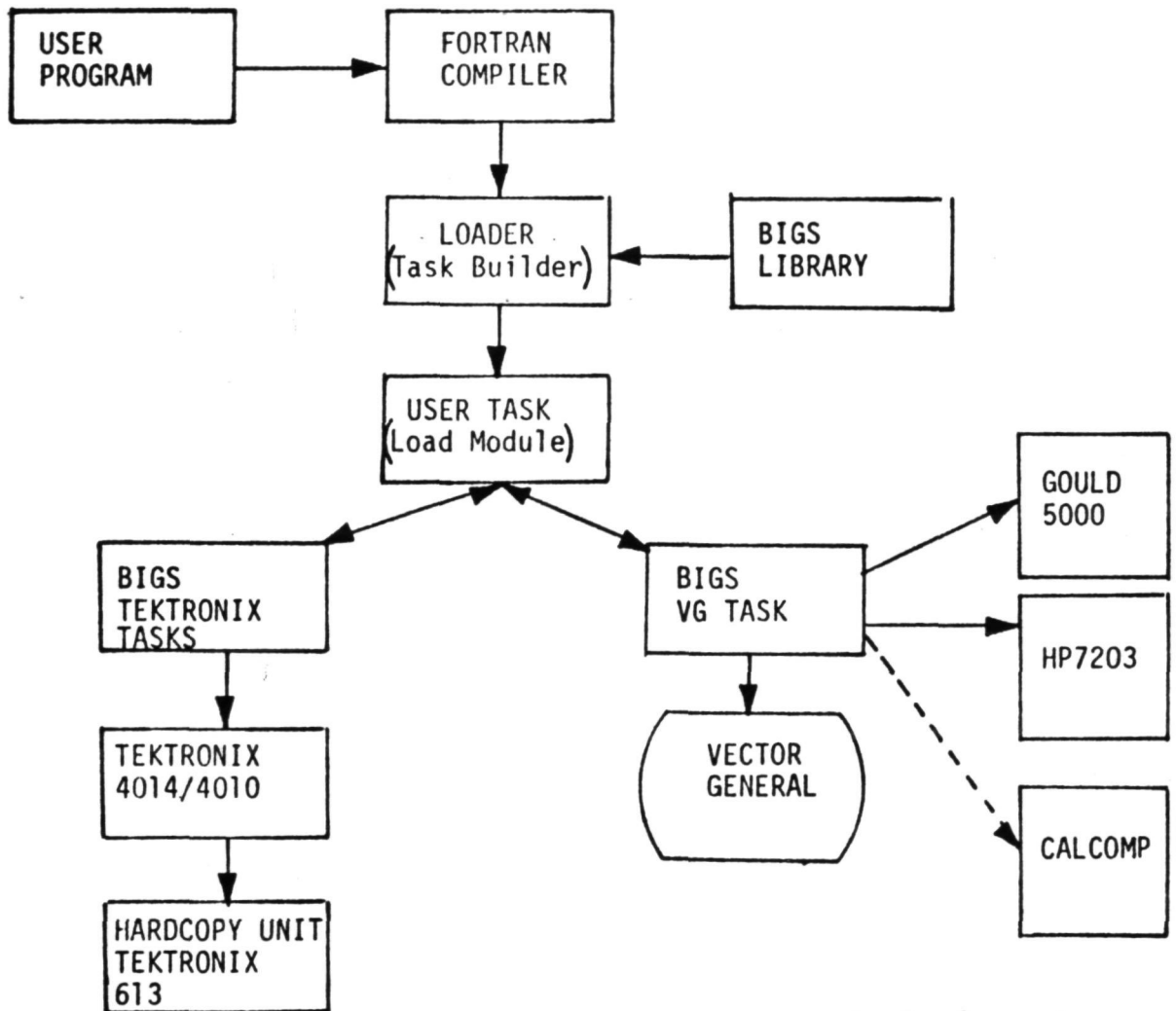


Figure 10.- Propulsion Computer System PDP 11/50.



- - - - - in development

Figure 11.- BIGS Mini Configuration - PDP 11/50.

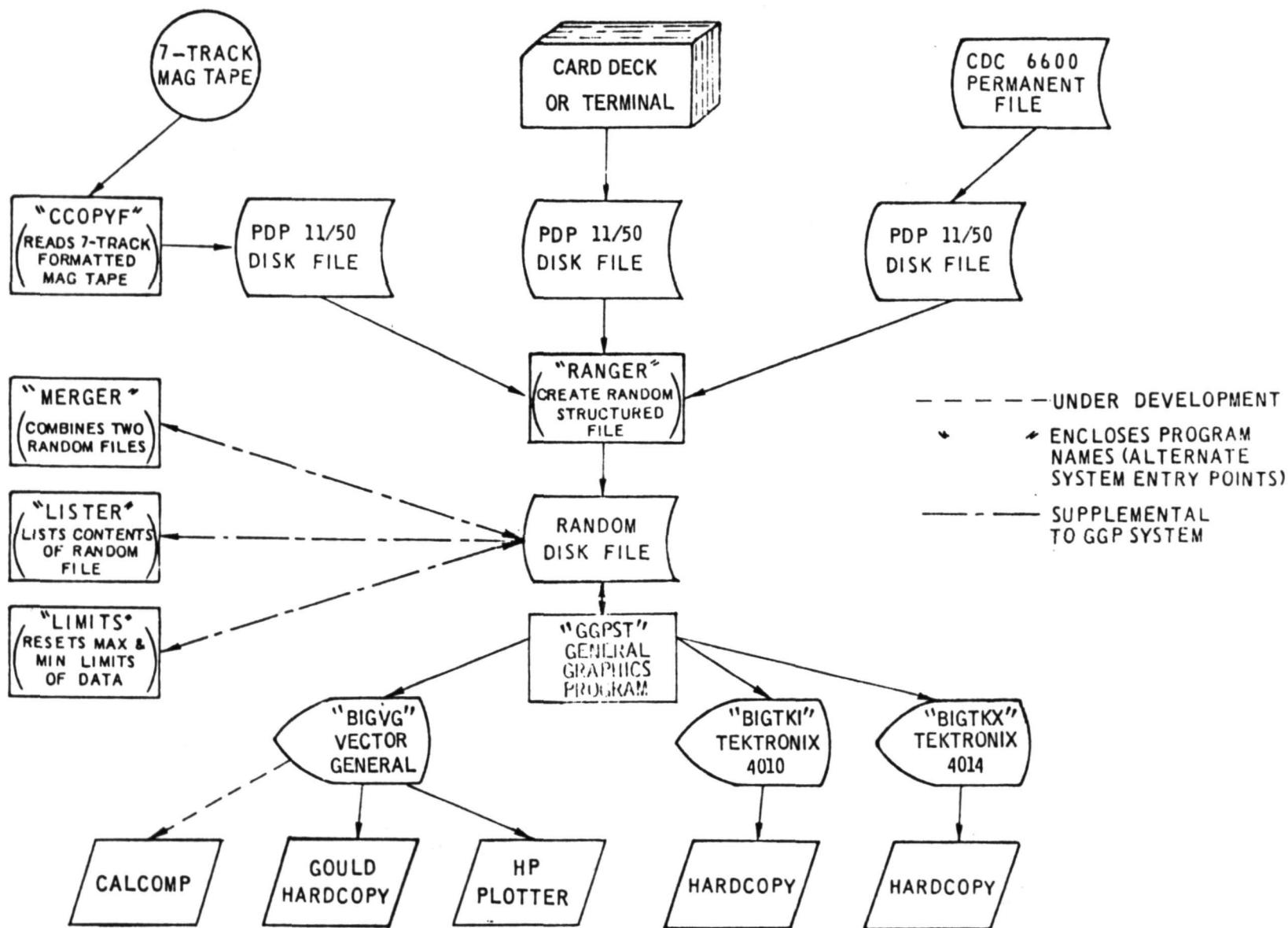
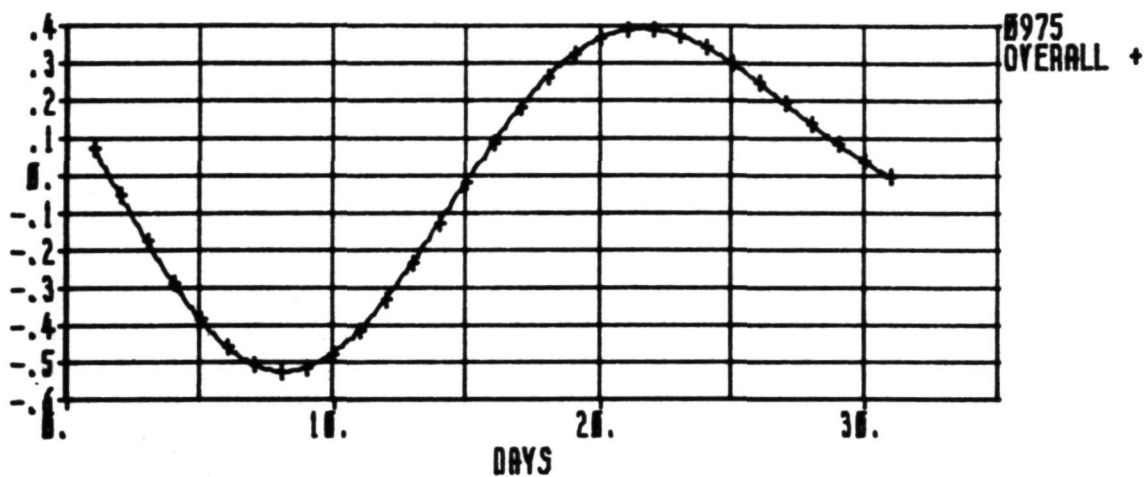


Figure 12.- GGP System.

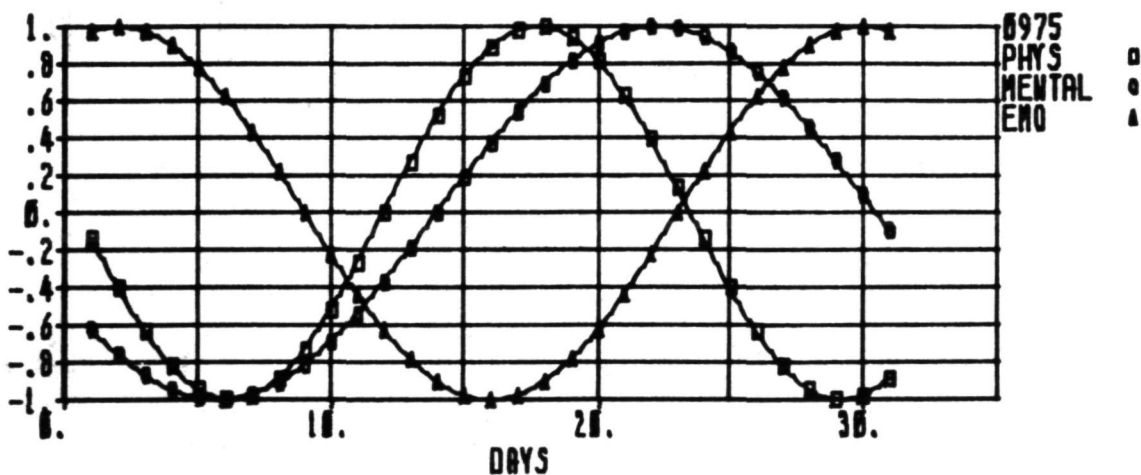


# DAYS VERSUS PHYS MENTAL EMO OVERALL

OVERALL



PHYS MENTAL EMO

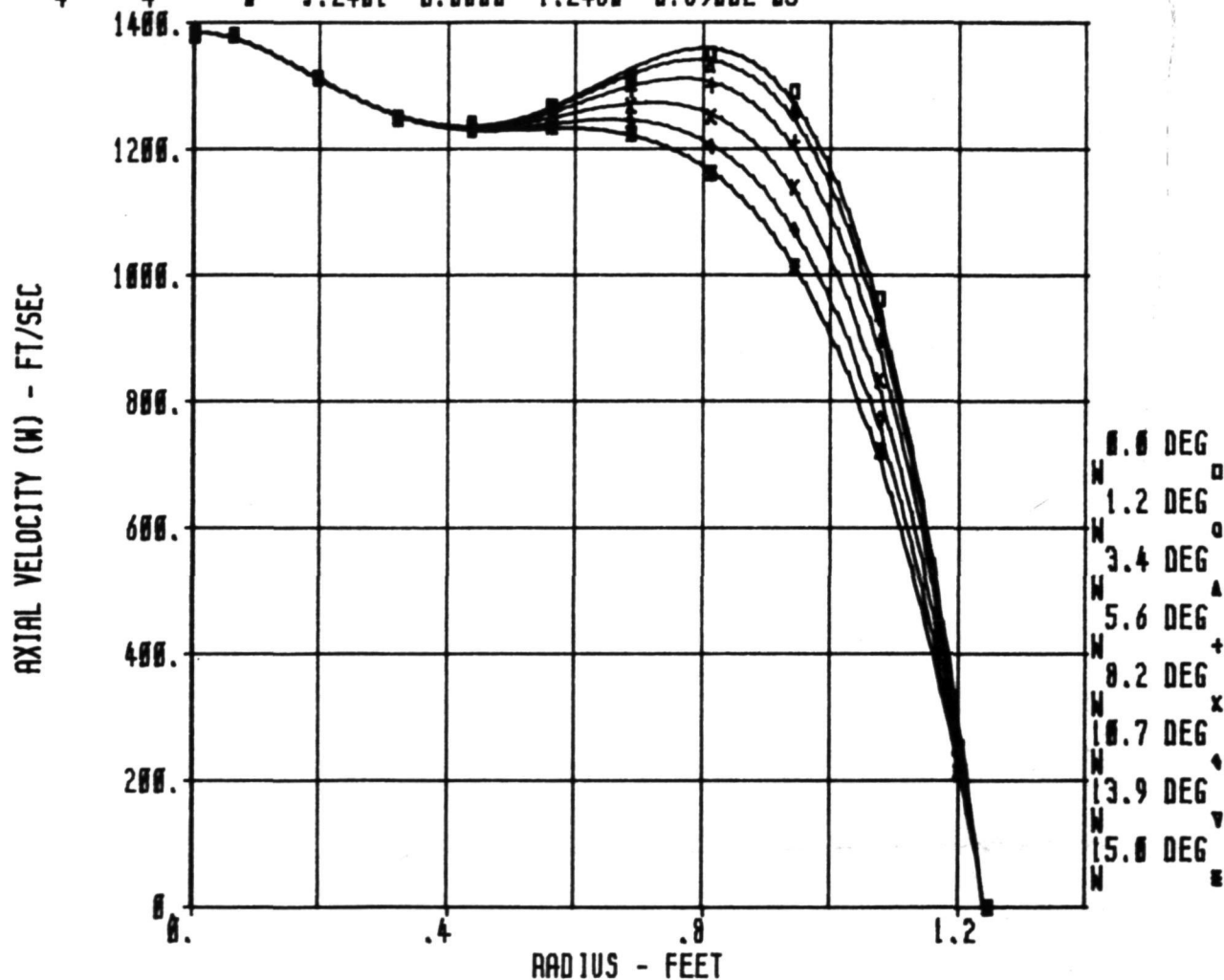


11-SEP-75 11:01:17

Figure 13.- Multiple Plot.

# FLOW FIELD CALCULATIONS DOWNSTREAM OF A LOBED MIXER NOZZLE

CASE 4 PLANE 4 1STEP 0 ZU 5.2401 R1 0.0000 R0 1.2460 PBAR 0.6900E 05



04-SEP-75 13:39:03

Figure 14.- Typical Display Plot.

## SELECT OPTION

DENSITY  
GRID TYPE  
FIX SCALE  
UNFIX SCALE  
GRID  
NO GRID  
NO AXES, GRID  
LEGEND  
NO LEGEND  
CHG SYMBOLS  
CHG POINT  
DEFAULT TYPE  
CHG TITLE  
CROSS PLOT  
COLOR  
CHG X-Y NAME  
CHG RUN NAME  
MANIPULATE  
COMBINE RUNS  
CAV FIT OPTS  
CAV FIT TYPE  
CONTOUR PLOT  
ORDER RUNS  
RUN AVERAGES  
SKIP POINTS  
TIME  
HP PLOT  
MULTI-GRIDS  
OPT SAVE

Figure 15.- Example of Menus Available with GGP.

# A GRAPHICS APPROACH IN THE DESIGN OF THE

## DUAL AIR DENSITY EXPLORER SATELLITES

David S. McDougal

NASA Langley Research Center

### SUMMARY

A computer program has been developed to generate a graphics display of the Dual Air Density (DAD) Explorer satellites which aids in the engineering and scientific design. The program displays a two-dimensional view of both spacecraft and their surface features from any direction. The graphics have been an indispensable tool in the design, analysis, and understanding of the critical locations of the various surface features for both satellites.

### INTRODUCTION

The Dual Air Density Explorer Program will launch, in late 1975, two spherical satellites to measure the density and composition of the Earth's upper atmosphere. The two satellites will be covered with a number of perforations, solar cells, and other surface features. The locations of the uniformly distributed perforations relative to other surface features are important in order to maximize the experiment measurement accuracy. A graphics package has been incorporated into a perforation design program to aid in the analysis of the perforation pattern. This paper describes the graphics program which generates two-dimensional views of both satellites.

### DUAL AIR DENSITY EXPLORER PROGRAM

The basic objective of the Dual Air Density Explorer Program is to expand man's knowledge of the Earth's upper atmosphere by measuring the density, composition, and vertical structure of the atmosphere on a global scale. Two spacecraft (fig. 1) will be launched in November 1975 by a single Scout vehicle into the same polar orbital plane. The two satellites will have perigees of 350 km and 700 km and apogees of 1500 km. Perforations uniformly cover each sphere surface so that the same amount of atmospheric constituents enters the hollow sphere regardless of satellite orientation. The sphere enclosure itself is an integral part of a highly sensitive mass spectrometer system which is located inside the satellite. Thus, the mass spectrometer measurements are

independent of satellite orientation -- which eliminates the need for an attitude control system. Also, in-flight calibration of the mass spectrometer system will be performed using drag data and on-board gas standards.

#### Lower Satellite

The lower satellite is a rigid .76 m (30 in.) sphere .001 m thick made of 6061 aluminum (fig. 2). Its main surface features are 85 holes (either .016 m or .018 m diameter) and 36 solar cells (.152 m x .168 m rectangles whose surfaces are curved concentric with the sphere surface). The center locations of the holes and solar cells are based on a symmetric 122 point pattern of a dodecahedron. Certain holes and solar cells, however, are displaced slightly from the pattern to accommodate the physical size of the solar cells and other surface geometry factors. Other surface features are four antenna pods (.05 m diameter plates), four antennas (flexible rods .53 m long), and an instrument package backplate (a cylinder of .138 m diameter extending .018 m above the sphere surface).

#### Upper Satellite

The upper satellite is a 3.66 m (12 ft) inflatable sphere (fig. 3) similar to the Langley Research Center Air Density Explorer inflatable drag satellites. It is constructed with 40 gores. Each gore has an identical perforation pattern of 80 holes (.028 m diameter), located at the center of 80 equal areas. The satellite hole pattern is modified by placing 20 solar cells (.254 m x .203 m flat plate rectangles) uniformly on the sphere (at the vertices of a dodecahedron) and eliminating those holes which fall within the view of the solar cells. The resulting hole pattern consists of approximately 2380 holes. The ring of holes closest to one pole have been moved out of the line of sight of an instrument package backplate, which is not included in the graphics.

#### GRAPHICS PROGRAM

##### Graphics Objective and Uses

The objective of the graphics is to display a two-dimensional view of the spacecraft and their surface features as viewed from any direction. The graphics are used to:

- visually verify relative locations and dimensions of the various surface features
- identify areas of close proximity between features

- aid in the design, analysis, and understanding of the critical locations of the features
- produce recognizable pictures of the spacecraft.

### Graphics Approach

Initially, an equatorial coordinate system is set up with position on the surface of the spherical satellite represented by  $\alpha$  (right ascension) and  $\delta$  (declination). The right ascension is the distance measured along an arbitrary reference plane (equatorial plane) from an arbitrary reference point (vernal equinox). The declination is the distance measured perpendicular to the equatorial plane.

A two-dimensional picture of the satellite and its surface features is calculated for any given view. A particular surface feature is drawn by representing its various edges or facets as a set of data points ( $\alpha, \delta$ ) that are plotted as a group. For example, the edge of a solar cell is drawn in the following fashion.

The location and dimensions of the solar cell are used to determine the coordinates of the solar cell's edge ( $\alpha, \delta$ ) in the equatorial coordinate system. The coordinates of the given view vector are also calculated in this coordinate system. The view vector coordinates are used as the basis of a new equatorial coordinate system, with the new vernal equinox aligned along the view vector (fig. 4). The solar cell edge coordinates are first rotated into the new coordinate system and then transformed into a rectangular coordinate system, whose Z axis is aligned along the view vector. The X-Y plane defines the plane of the two-dimensional picture. The X-Y coordinates of the solar cell edge, with positive Z coordinates, are then plotted directly (fig. 4). (Negative Z coordinates correspond to features on the "back side" of the spacecraft.)

The remaining edges and other facets of the solar cell (grid lines on the upper face) are drawn in this manner. The process is repeated for the other surface features (holes, solar cells, etc.). For pictures of the spacecraft viewed from another direction, the coordinates of the new view vector are calculated and the method repeated. The equations describing the calculation of the coordinates of the view vector and a point on the sphere surface in the new equatorial coordinate system and rectangular coordinate system are given in the Appendix.

### RESULTS

A graphics picture of the lower .76 m satellite is shown in figure 5. The view vector is located at the vernal equinox ( $\alpha = 0, \delta = 0$ ). The individual holes are numbered as an aid in identifying their relative location with respect to the solar cells. The grid lines on the solar cells depict the strings of

individual chips on each cell. Also, the thickness of the cells is graphically drawn (originally used to identify the "shadow" effect of the solar cells on the holes with respect to the velocity vector). Also shown are the four antenna pods and antennas.

A picture of the upper 3.66 m satellite is shown in figure 6 with the view vector at the vernal equinox ( $\alpha = 0$ ,  $\delta = 0$ ). The alignment of the hole pattern along the length of each gore is readily apparent. Holes that lie within the field of view of each solar cell have been eliminated. As can be seen, both figures give a recognizable picture of the two spacecraft and their surface features.

As an aid in understanding how specific hole locations affected the hole uniformity design, the graphics was originally used to focus on particular orientations. While an actual description of the hole uniformity design is beyond the scope of this paper, some of the views generated by the graphics as the satellite rotates about a spin axis are shown in figure 7 for the lower satellite and figure 8 for the upper satellite. The view vectors vary along a plane inclined  $45^\circ$  to the satellite equatorial plane. The right ascension and declination of the view vectors are given in the upper right hand corner of each picture. The figures represent how a spinning satellite would appear to an observer over a quarter rotation.

#### CONCLUSION

Both sets of figures give recognizable views of the spacecraft and their surface features as viewed from any direction. The graphics was an invaluable aid in verifying the relative locations and dimensions of the various surface features, identifying areas of close proximity between surface features, and understanding the critical locations of the hole patterns to maintain the uniformity criteria.

## APPENDIX

The coordinates of the view vector ( $\alpha_v$ ,  $\delta_v$ ) are calculated from

$$\sin \delta_v = \sin i \sin \theta \quad (1)$$

$$\cos (\alpha_v - \Omega) = \cos \theta / \cos \delta_v \quad (2)$$

where  $i$  = inclination of spin plane relative to the equator

$\theta$  = angle measured along spin plane from the equator

$\Omega$  = ascending node, or distance along equator from vernal equinox ( $\alpha = 0$ ) to intersection of spin plane with equator

The inclination ( $\psi$ ) of the new equatorial plane is calculated from

$$\cos \psi = \cos \delta_v \sin \alpha_v / \sin \alpha_x \quad (3)$$

$$\cos \alpha = \cos \delta_v \cos \alpha_v \quad (4)$$

where  $\alpha$  is the angle from the vernal equinox to the view vector.<sup>x</sup>

The coordinates of a location on the sphere surface in the new equatorial system ( $\alpha'$ ,  $\delta'$ ) are calculated from

$$\sin \delta' = \cos \psi \sin \delta - \sin \psi \cos \delta \sin \alpha \quad (5)$$

$$\tan (\alpha' + \alpha_x) = (\sin \psi \sin \delta + \cos \psi \cos \delta \sin \alpha) / \cos \alpha \cos \delta \quad (6)$$

where  $\alpha$ ,  $\delta$  are coordinates of location in old equatorial system

The coordinates of a location on the sphere in a rectangular coordinate system (with Z axis aligned along view vector, X axis aligned along new equatorial plane, and Y axis aligned perpendicular to both) are calculated from

$$X = r \sin \alpha' \cos \delta' \quad (7)$$

$$Y = r \sin \delta' \quad (8)$$

$$Z = r \cos \alpha' \cos \delta' \quad (9)$$

where  $r$  is the distance from the sphere center to the point on the sphere surface.



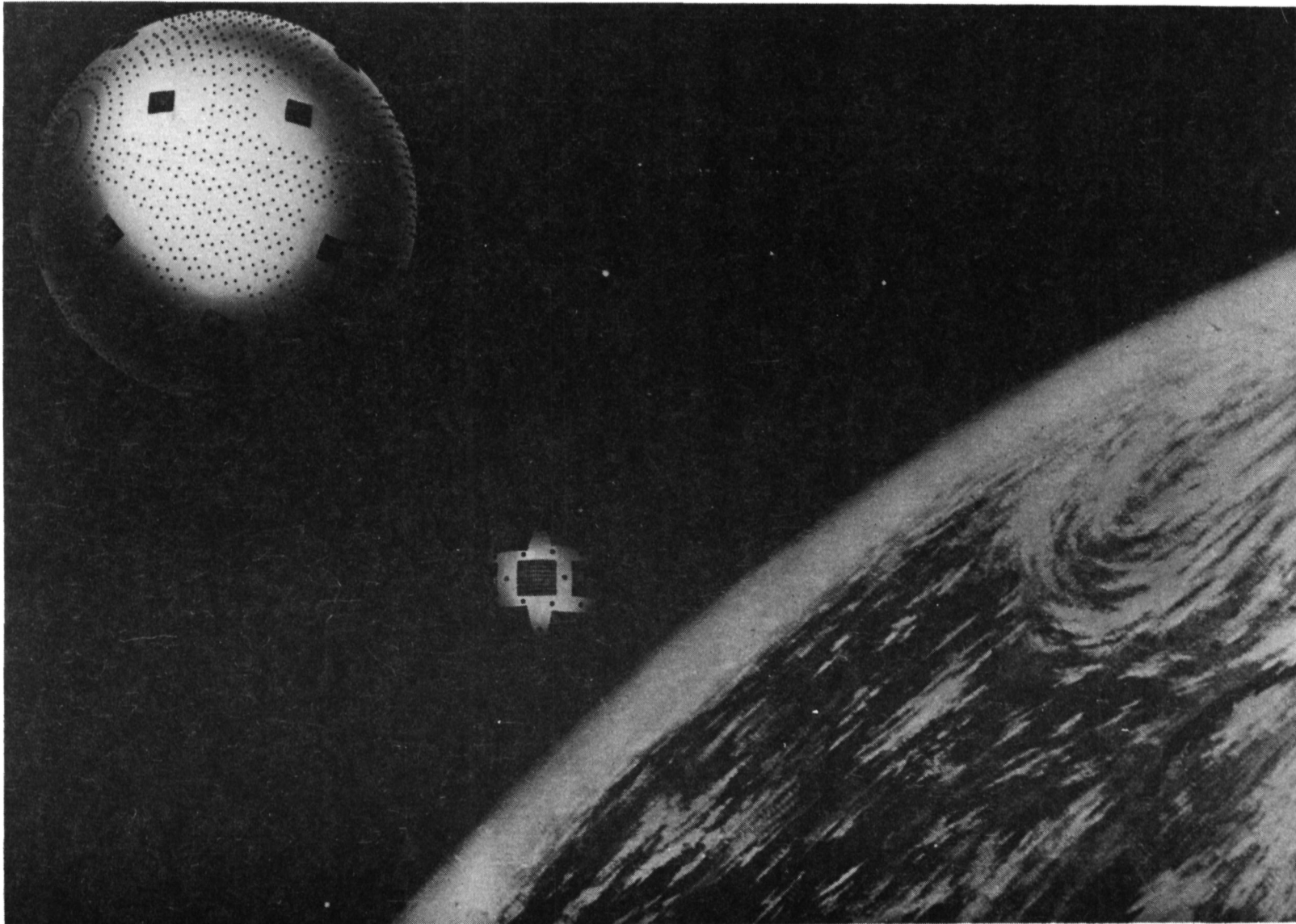


Figure 1.- Schematic concept of Dual Air Density Explorer satellites.

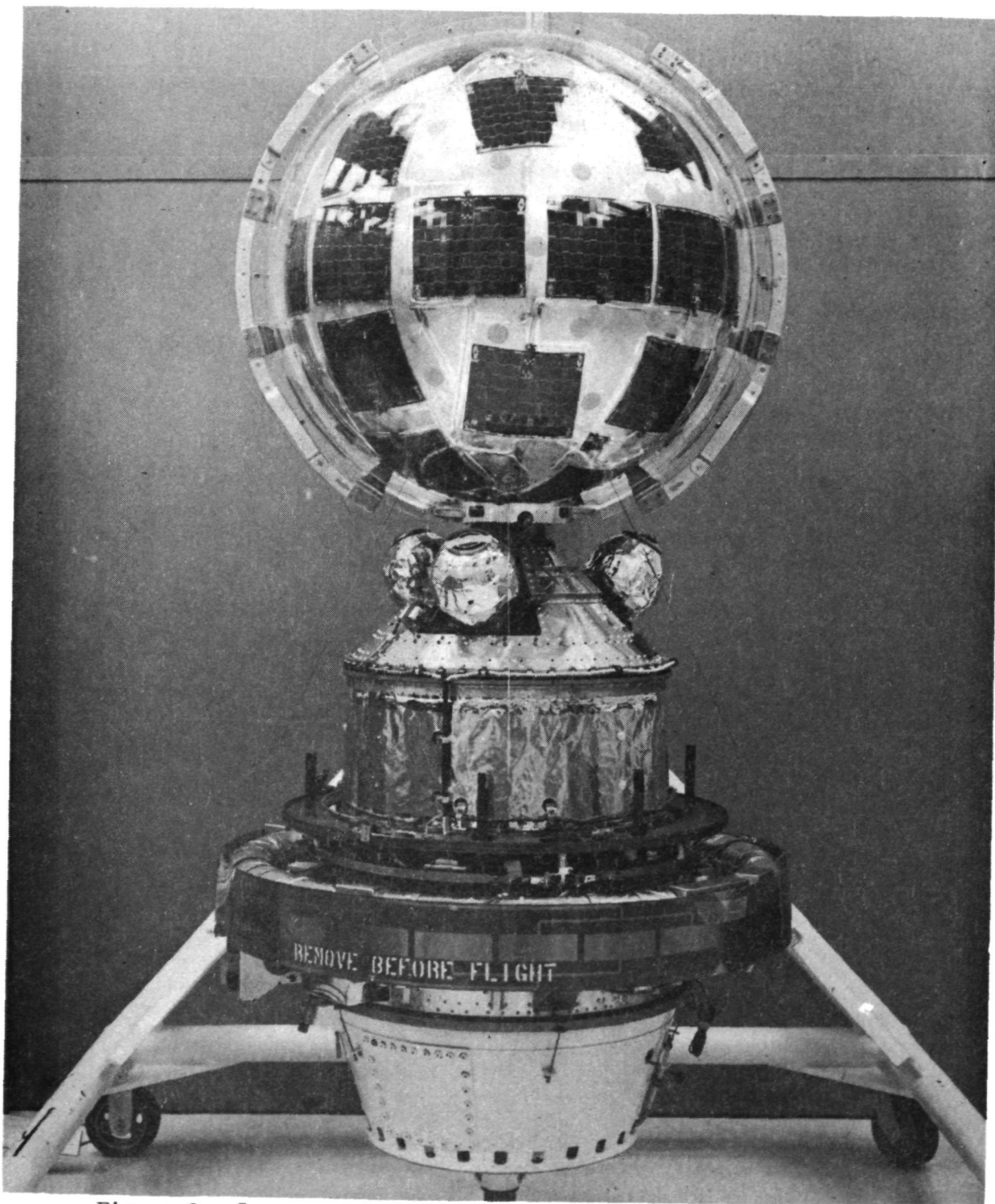


Figure 2.- Lower satellite in launch configuration showing the .76 m satellite, orbit booster package, 3.66 m satellite storage cannister, and inflation ring.

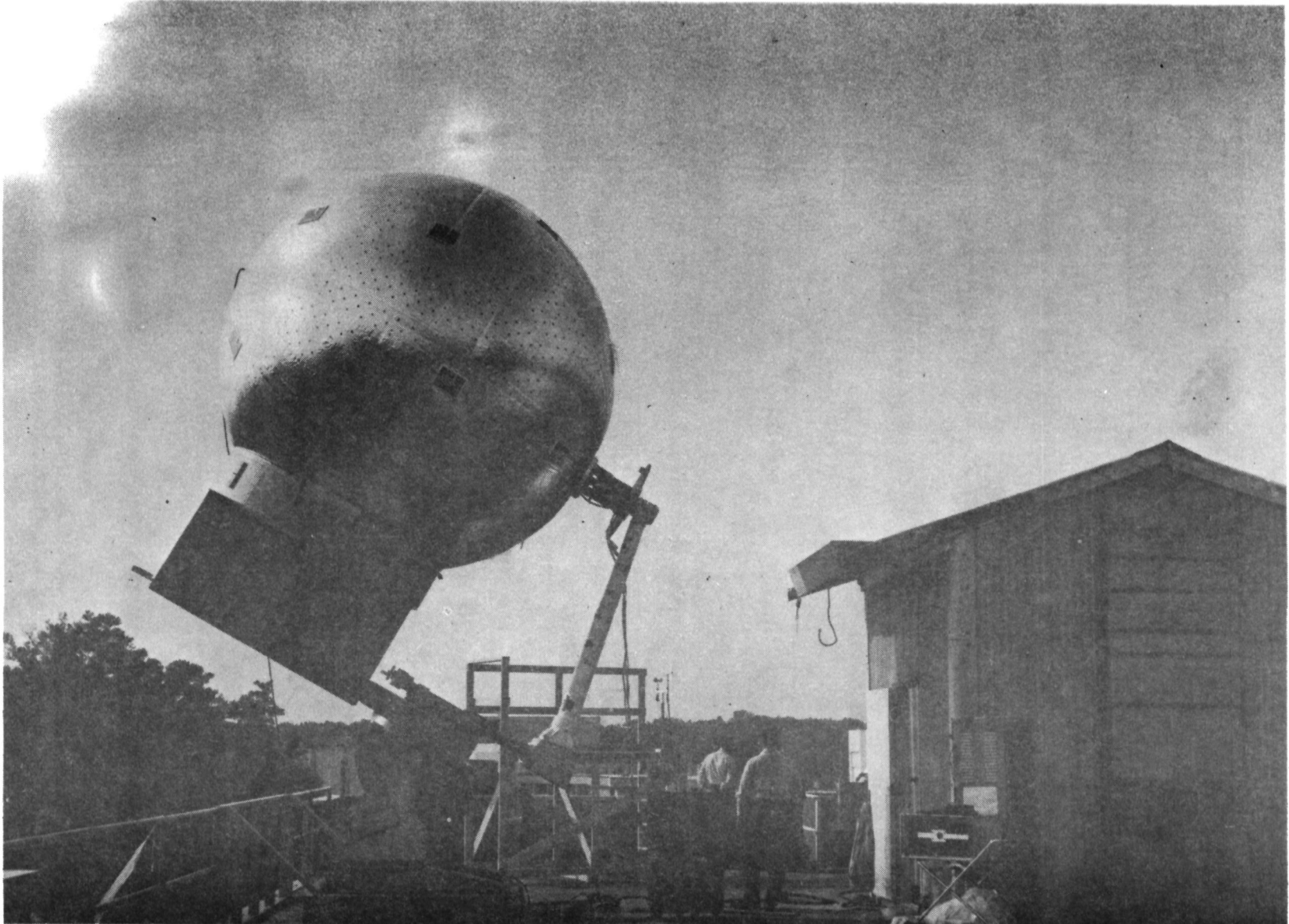
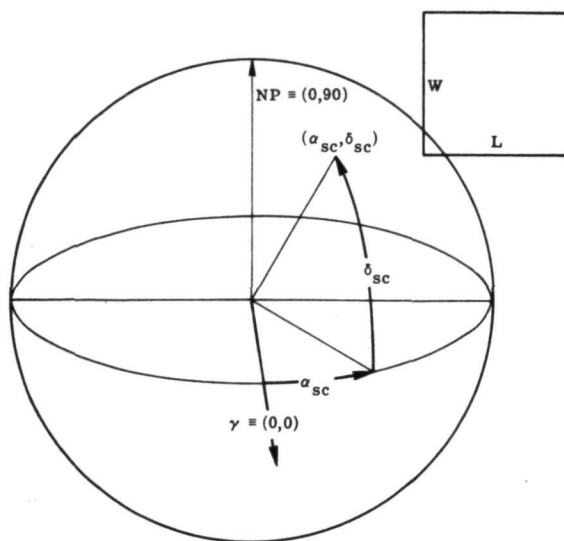
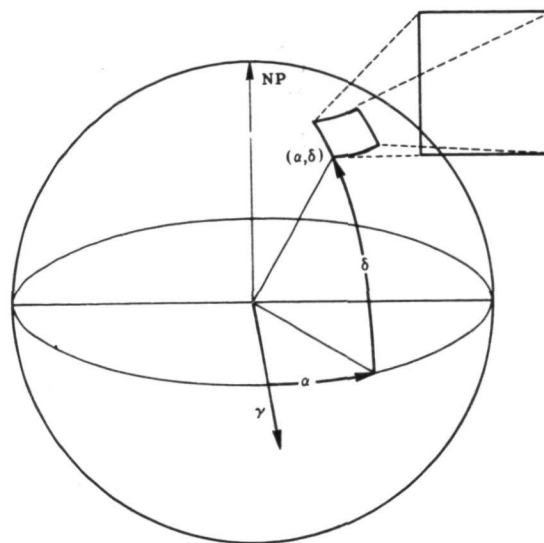


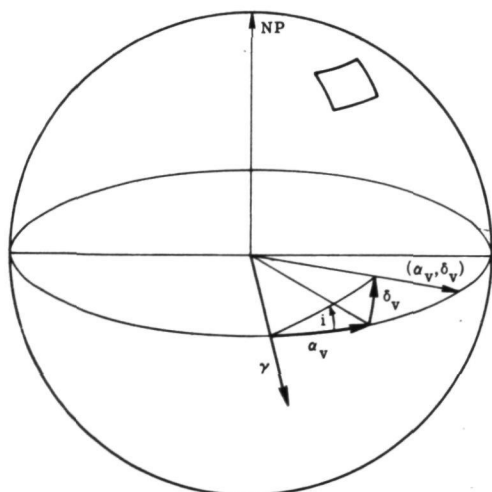
Figure 3.- Upper satellite fully inflated during antenna pattern test.



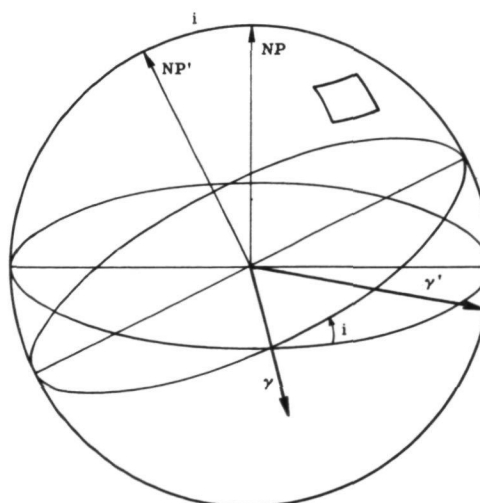
From the location of the center of each solar cell  $(\alpha_{sc}, \delta_{sc})$   
and dimensions  $(L, W)$ , . . .



calculate the equatorial coordinates of edge of solar cell  $(\alpha, \delta)$ .



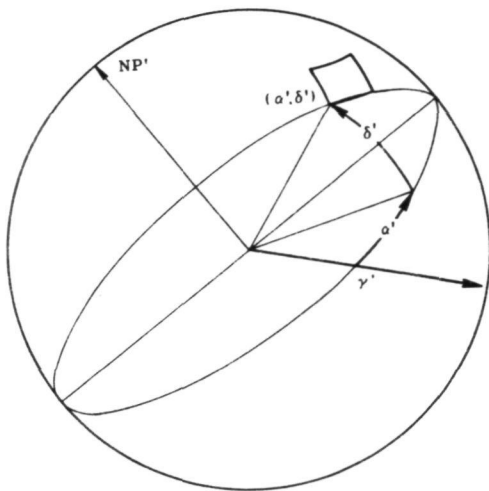
From the coordinates of the view vector  $(\alpha_v, \delta_v)$ , . . .



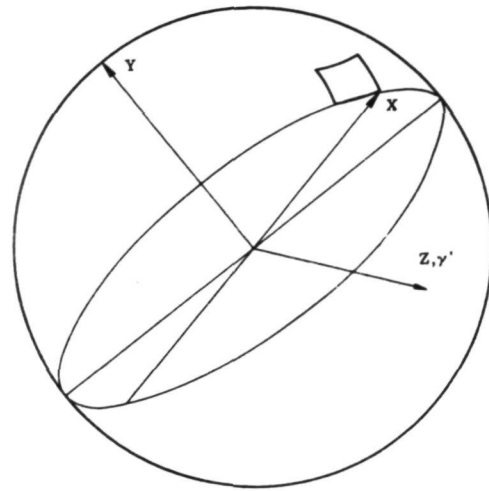
rotate into a new equatorial coordinate system

(with view vector at  $\gamma'$ ), and . . .

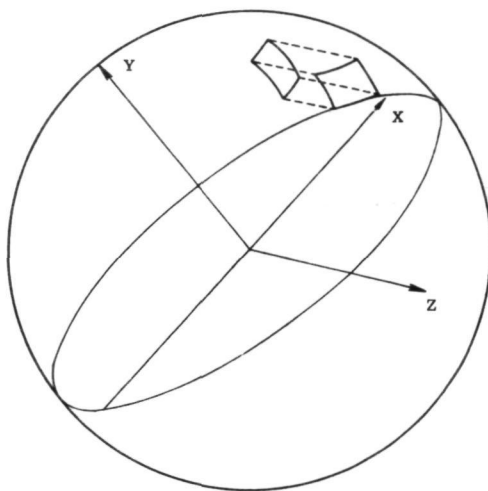
Figure 4.- Schematic diagrams describing graphics method.



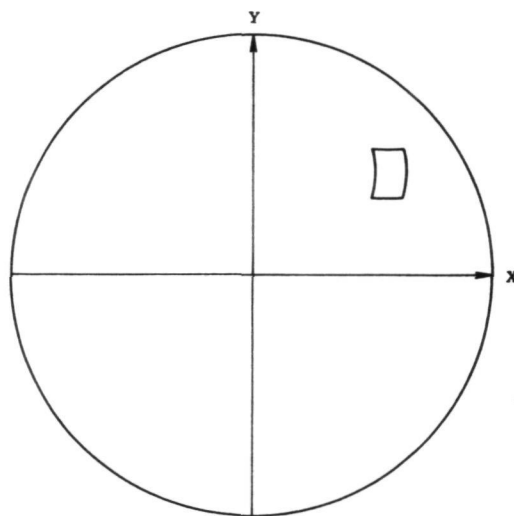
calculate new equatorial coordinates of edge of solar cell ( $\alpha'$ ,  $\delta'$ ).



Transform from the new equatorial coordinate system into an XYZ rectangular coordinate system (with Z axis at  $\gamma'$ ), and . . .



calculate rectangular coordinates of edge of solar cell (XYZ).



Plot directly the XY coordinates of edge of solar cell.

Figure 4.- Concluded.

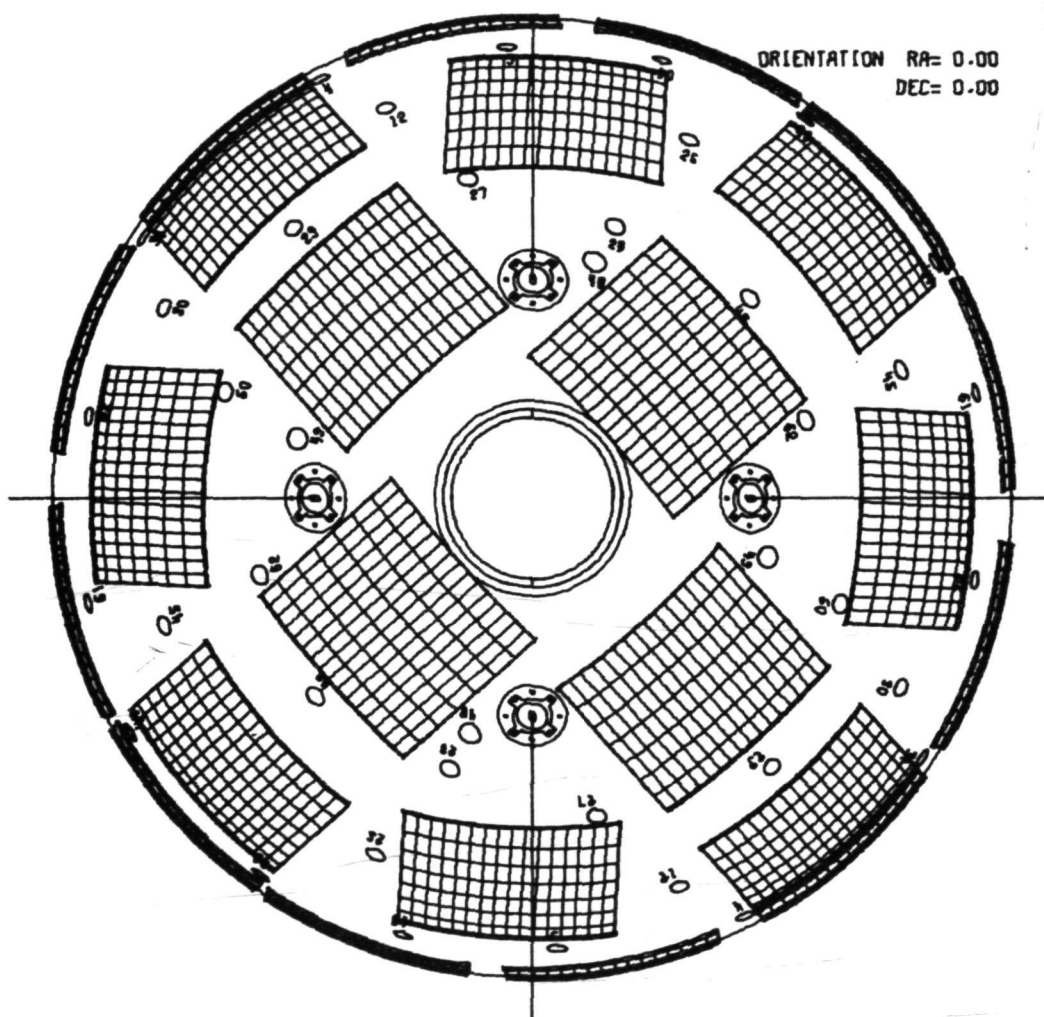


Figure 5.- Graphics plot of lower satellite showing holes, solar cells, antenna pods, antennas, and instrument package backplate.

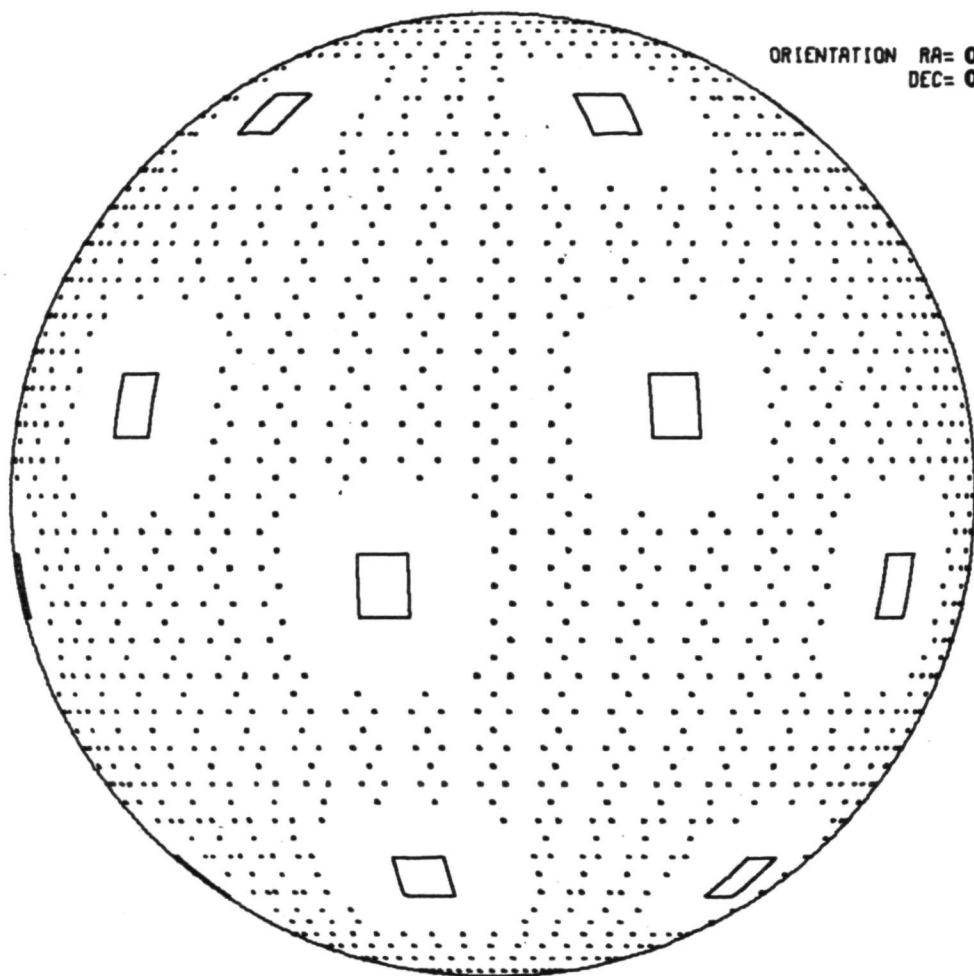


Figure 6.- Graphics plot of upper satellite showing  
holes and solar cells.



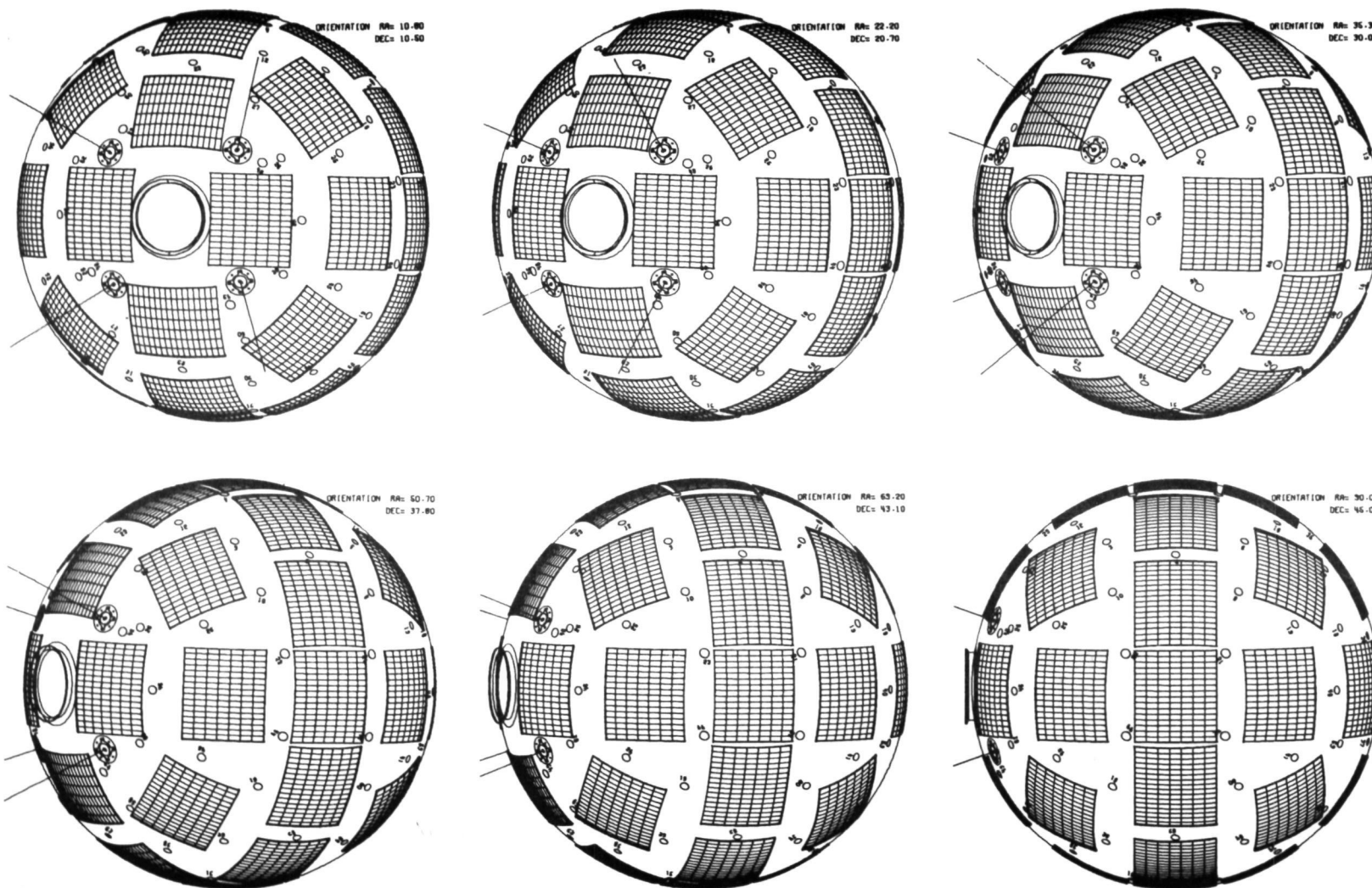


Figure 7.- Graphics plots of lower satellite over quarter rotation about a spin axis inclined  $45^{\circ}$  to the equator.



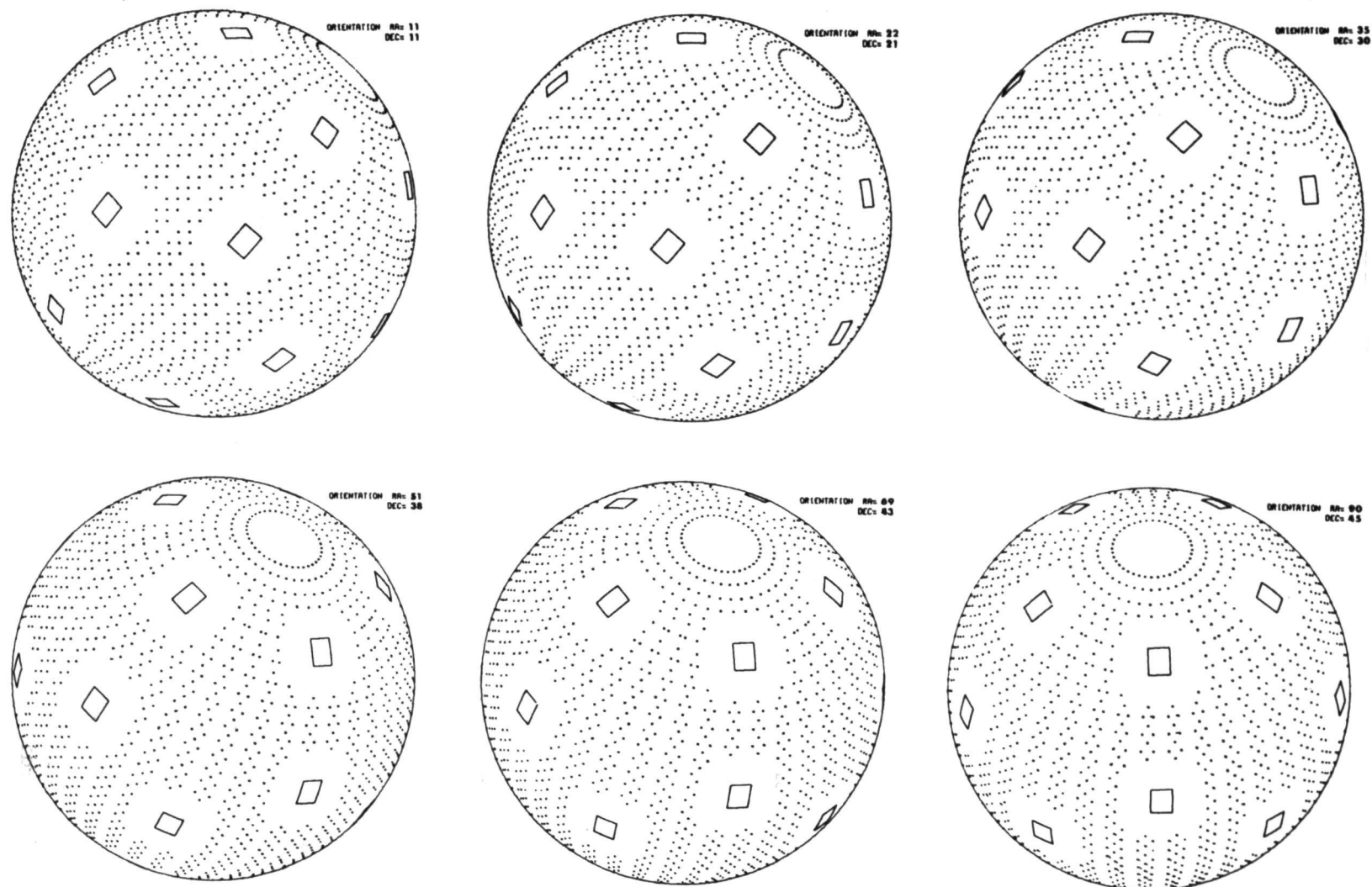


Figure 8.- Graphics plots of upper satellite over quarter rotation about a spin axis inclined  $45^{\circ}$  to the equator.

A COMPUTER PROGRAM FOR FITTING  
SMOOTH SURFACES TO THREE-DIMENSIONAL  
AIRCRAFT CONFIGURATIONS

Charlotte B. Craidon and Robert E. Smith, Jr.

NASA Langley Research Center

SUMMARY

This paper describes a computer program developed at the Langley Research Center to fit smooth surfaces to the component parts of three-dimensional aircraft configurations. The resulting equation definition of an aircraft numerical model is useful in obtaining continuous two-dimensional cross section plots in arbitrarily defined planes, local tangents, enriched surface plots and other pertinent geometric information.

At Langley Research Center and other government and industrial facilities, the geometry organization used as input to the program has become known as the Harris Wave Drag Geometry. An early version of this point surface definition was introduced in a NASA TM X-947. (See ref. 1.)

INTRODUCTION

An important part of the design, analysis, and construction of aerospace vehicles, as well as ships, automobiles, etc., is the construction of surface models for the required study. Mathematical models are the most useful because of their versatility and exactness of definition. Many more different mathematical models may be analyzed by a computer program than would be possible by any other method within a given time frame.

A set of planes defined by points to approximate a curved surface is the simplest type of mathematical model. A large number of points on the surface must be given to achieve an accurate definition using a set of planes.

The surface fitting technique used in this paper is based on approximating complex surfaces by piecing together smaller surface "patches." The surface equations were conceived and developed by Steven Coons. The term "Coons' Patch" is associated with this piecewise definition. (See ref. 2.) The computer program under discussion uses the bicubic form of the patch equation. (See ref. 3.) This form of the patch equation requires 48 parameters consisting of the coordinate position of the corners of the patch (12 parameters), the derivatives of the variables at the corners with respect to the parametric

independent variables (24 parameters), and the cross derivatives with respect to the parametric variables (12 parameters).

Although patch surface definition has been applied in many computer programs, the process of determining the derivatives has been varied. One process has been to apply phantom points near the corner points to approximate the derivatives by finite differencing. Many programs, including the one described herein, set the cross derivatives equal to zero. The LRC surface program uses parametric cubic spline functions to approximate the derivatives of the space variables with respect to the parametric independent variables.

The organization of the geometry input data to the program is identical to the geometry used for several standardized aerodynamic analysis computer programs. (See ref. 4.) By using the same input format throughout the design and analysis procedure, the designer can be assured of a consistent numerical model.

It should be pointed out that the program is not limited to aircraft configurations since a component part such as a fuselage can be some other object. Also, it is a simple matter to substitute an alternate data input format if this might prove more convenient.

Current program options that are available are to present on an XY plotter or on a cathode ray tube orthographic projections and cross section or contour plots. The patch equations are used to compute the required vectors.

A complete description of the program and its usage is documented in NASA TM X-3206. (See ref. 5.)

#### SYMBOLS

A,B,C,D	parametric cubic spline coefficients
a,b,c,d	plane equation coefficients
$\bar{B}$	boundary matrix
G	matrix equation of the solution of a patch and a plane
L	chord length
M	blending function matrix
P	a vector whose components are functions of t
S	component of surface patch equation
t	independent variable in cubic equation

$u, w$	independent variables in patch equation
$V$	a vector whose components are functions of $u$ and $w$
$x, y, z$	coordinates of a point
$\theta$	pitch angle
$\phi$	roll angle
$\psi$	yaw angle

## PROBLEM DESCRIPTION AND METHOD OF SOLUTION

A functional definition of an aircraft numerical model is useful for obtaining pertinent geometric information for use in aeronautical analysis programs or for building models for wind-tunnel testing. A computer program has been developed to provide this functional definition in surface patch equations.

The organization of the input data for the program is broken down into various component parts of an aircraft configuration such as wing, fuselage, nacelle, vertical and horizontal tails. A typical input configuration is given in figure 1. Sufficient data points with no abrupt changes in curvature must be supplied as the modeling technique is designed to approximate a smooth surface.

A surface patch is a portion of surface bounded by space curves and is a function of two variables,  $u$  and  $w$ , which can have only the values between 0.0 and 1.0. It is necessary to supply 48 parameters for each patch equation. A  $4 \times 4$  boundary matrix is constructed for each component of the patch equation ( $X, Y, Z$ ). It is called a boundary matrix because the parameters are functions of the patch boundaries: corner points, corner derivatives and cross derivatives. The cross derivatives are set equal to zero in this application.

The boundary matrix is then pre- and postmultiplied by constant-valued blending functions and stored for further use in this form. The blending functions provide a blending effect from one surface patch to an adjoining patch. Each surface patch is represented in exactly the same matrix form which proves to be very convenient for computer processing. The basic patch equations are given in figure 2.

The corner derivatives for the surface patch equations are obtained by using a three-dimensional parametric cubic spline curve fit technique. The coordinate points describing the curve are expressed as cubic functions of one variable. Consecutive points in the  $u$ -direction are fitted in this manner and also in the  $w$ -direction.

In the curve fit method, a series of adjacent polynomial segments between

each given point is used to represent the curve. The adjacent cubic segments are related by setting the second derivatives equal at the common points.

A matrix equation is developed where the slopes are the unknowns. The second derivatives at the first and last points on a curve are made equal to zero since there are two less equations than unknowns. Since the matrix equation is tridiagonal, the Thomas Algorithm, which is equivalent to Gaussian Elimination without pivoting, is used for the solution. The Thomas Algorithm is a rapid, recursive method for solving a tridiagonal system of equations. Figure 3 shows the basic equations used.

Parametric cubic splines were chosen because of their several advantages. Parametric curves are not sensitive to infinite slopes, spline curves approximate smoothness satisfactorily, and cubics are the lowest order polynomial able to twist through space.

To create orthographic projections of the given body surface, angles of yaw, pitch, and roll must be supplied. The three-dimensional rotation equations are applied directly to the patch equations, where the rotated patch equation has only two components for the desired paper plane.

The surface plots may be enriched to any degree. Two equations in  $w$  may be obtained by giving  $u$  a value between 0.0 and 1.0. These equations may be solved by giving  $w$  values between 0.0 and 1.0 for additional body surface lines in the  $w$ -direction. Similarly, two equations in  $u$  may be obtained by giving  $w$  a value between 0.0 and 1.0, and solving by varying  $u$  from 0.0 to 1.0 for additional curves in the  $u$ -direction.

A partial "hidden line" test may be used to delete points that face away from the viewer. A surface normal vector is computed from the rotated and projected patch equation at each requested point. The computed point is visible and plotted if the surface vector is positive. If the surface vector is negative, the point is not plotted. Figure 4 gives the basic equations for the surface plots and figure 5 presents some examples.

The simultaneous solution of the patch equations and the equation of a plane is used to provide cross section or contour plots. The cutting plane is defined by three points and may be at any orientation. The result is a  $4 \times 4$  matrix composed of constant elements and may be evaluated for an equation in the two variables  $u$  and  $w$ . The problem then becomes that of finding the zeros of a cubic polynomial in one of the parametric variables. Since there are three zeros, the program has logic in it to pick one of the points. If an error is made, it normally stands out immediately. The points defining the curve of intersection are then rotated and translated so that the intersection plane coincides with the YZ-plane of the paper. Figure 6 gives the basic equations for obtaining the cross section or contour plots and some typical output plots are given in figure 7.

## CONCLUDING REMARKS

A computer program has been written that fits smooth surfaces to the component parts of an aircraft configuration. The resulting surface patch equations can be useful anywhere an equation definition may be required. Program options currently provide plotting of the enriched surface at any desired viewing angle and the solution of a number of patches and a plane for generating cross section or contour plots through the given surfaces.

Although the program was written to accept a point surface definition of an aircraft as input, it is a simple matter to substitute another input format to describe another object.

The program has been used for on-line display on a cathode ray tube and to drive Gerber, Varian, and Calcomp plotting devices.

## REFERENCES

1. Harris, Roy V., Jr.: An Analysis and Correlation of Aircraft Wave Drag. NASA TM X-947, 1964.
2. Coons, Steven A.: Surfaces for Computer-Aided Design of Space Forms. MAC-TR-41 (Contract No. AF-33(600)-42859), Massachusetts Inst. Technol., June 1967. (Available from DDC as AD 663 504.)
3. Eshleman, A. L.; and Meriwether, H. D.: Graphic Applications to Aerospace Structural Design Problems. Douglas Paper 4650, McDonnell Douglas Corp., Sept. 1967.
4. Craidon, Charlotte B.: Description of a Digital Computer Program for Airplane Configuration Plots. NASA TM X-2074, 1970.
5. Craidon, Charlotte B.: A Computer Program for Fitting Smooth Surfaces to an Aircraft Configuration and Other Three-Dimensional Geometries. NASA TM X-3206, 1975.

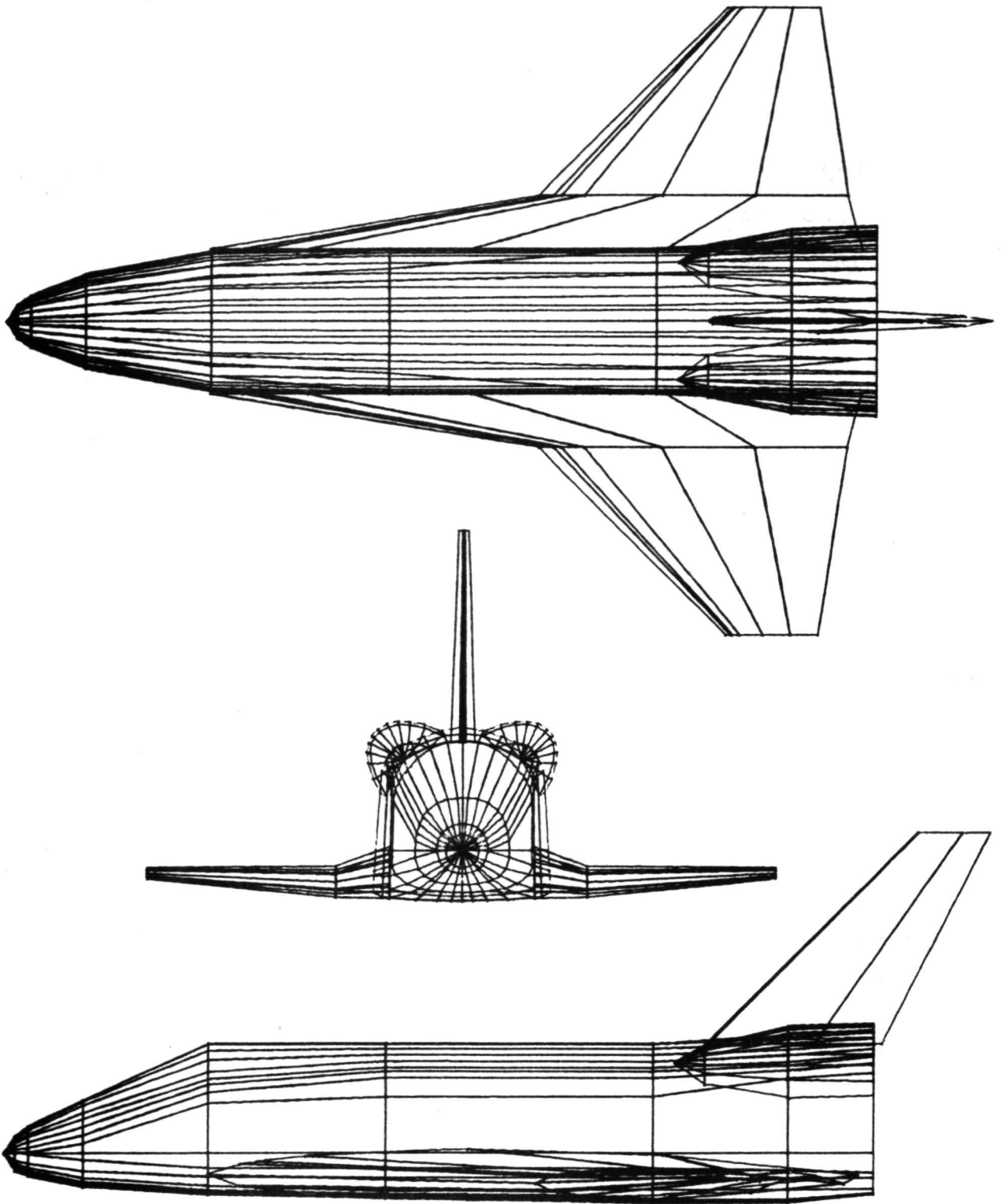
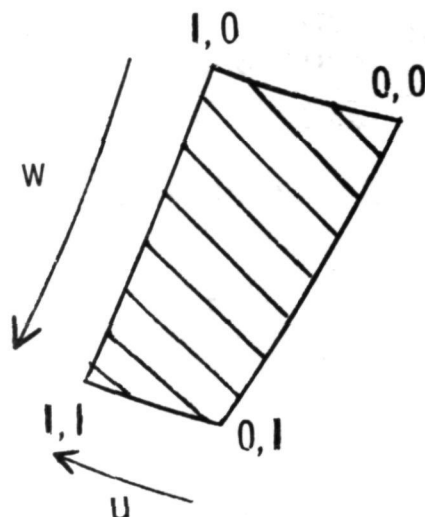
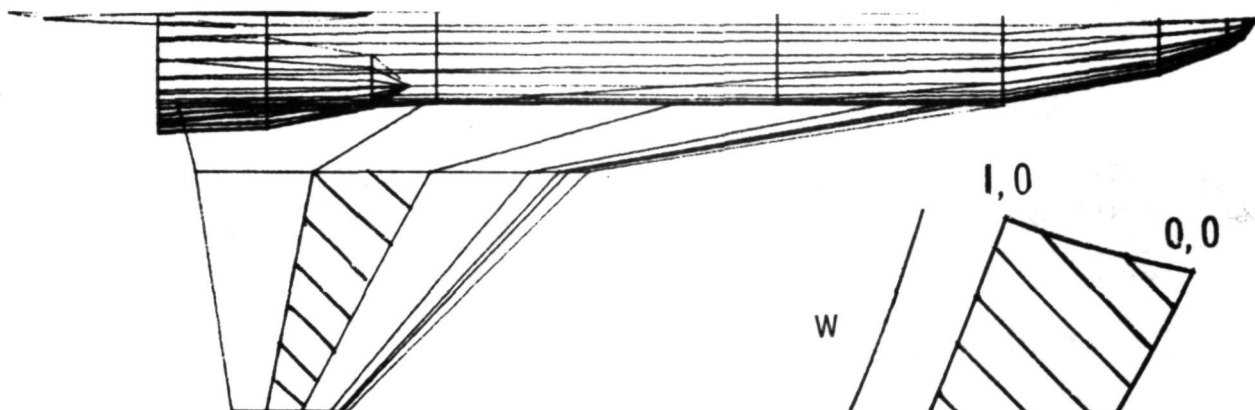


Figure 1. - A typical input configuration.



$$\begin{aligned} X(u, w) \\ Y(u, w) = V(u, w) = U \bar{B} M^t W^t = U S W^t \\ Z(u, w) \end{aligned}$$

where:

$$U = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}$$

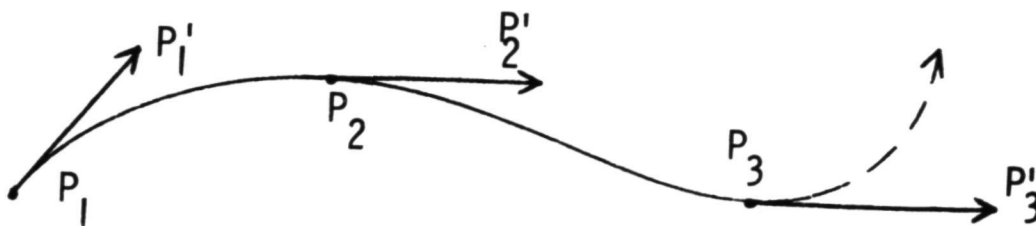
$$W = \begin{bmatrix} w^3 & w^2 & w & 1 \end{bmatrix}$$

$$M = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\bar{B} = \begin{bmatrix} V(0,0) & V(0,1) & V_w(0,0) & V_w(0,1) \\ V(1,0) & V(1,1) & V_w(1,0) & V_w(1,1) \\ V_u(0,0) & V_u(0,1) & V_{uw}(0,0) & V_{uw}(0,1) \\ V_u(1,0) & V_u(1,1) & V_{uw}(1,0) & V_{uw}(1,1) \end{bmatrix}$$

Figure 2. - Surface patch equation.





Cubic Equation:

$$X(t) =$$

$$Y(t) = P(t) = At^3 + Bt^2 + Ct + D$$

$$Z(t) =$$

Cubic Coefficients in Terms of End Points and Slopes (for one segment):

$$A_1 = \frac{2(P_1 - P_2)}{L_1^3} + \frac{P'_1}{L_1^2} + \frac{P'_2}{L_1^2}$$

$$B_1 = \frac{3(P_2 - P_1)}{L_1^2} - \frac{2P'_1}{L_1} - \frac{P'_2}{L_1}$$

$$C_1 = P'_1$$

$$D_1 = P_1$$

System of Equations to Solve for Unknown Slopes:

$$\begin{bmatrix} L_2 & 2(L_1 + L_2) & L_1 & \bigcirc \\ & L_3 & 2(L_2 + L_3) & L_2 \\ & & \bigcirc & \bullet \\ & & \bullet & \bullet \end{bmatrix} \times \begin{bmatrix} P'_1 \\ P'_2 \\ \vdots \\ P'_n \end{bmatrix} = \begin{bmatrix} f(L_1, L_2, P_1, P_2, P_3) \\ f(L_2, L_3, P_2, P_3, P_4) \\ \vdots \end{bmatrix}$$

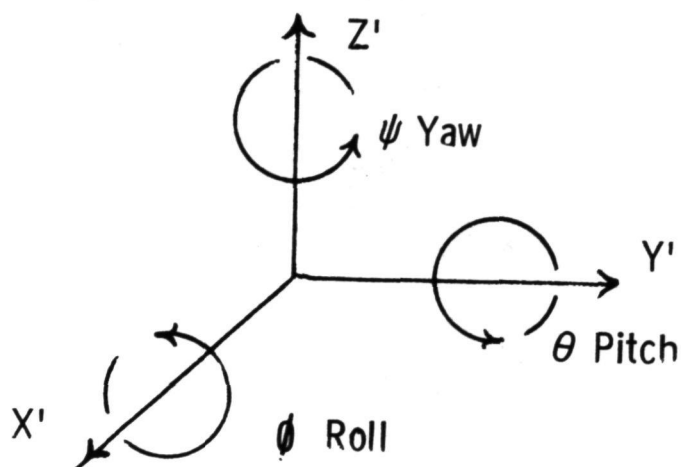
Figure 3. - Parametric cubic spline curve fitting.

Two Component Rotated Patch Equation:

$$S'_x = S_x f_1(\phi, \theta, \psi) + S_y f_2(\phi, \theta, \psi) + S_z f_3(\phi, \theta, \psi)$$

$$S'_y = S_x f_4(\phi, \theta, \psi) + S_y f_5(\phi, \theta, \psi) + S_z f_6(\phi, \theta, \psi)$$

where:



Enriched, Rotated Patch:

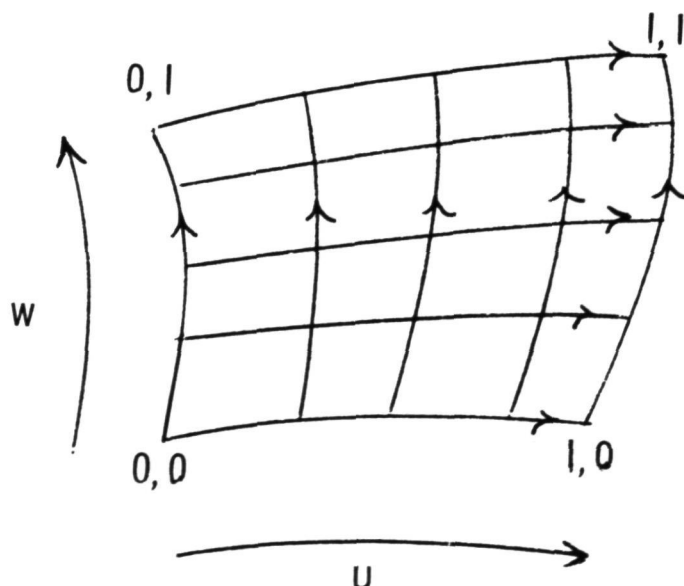


Figure 4. - Rotated and enriched patch.

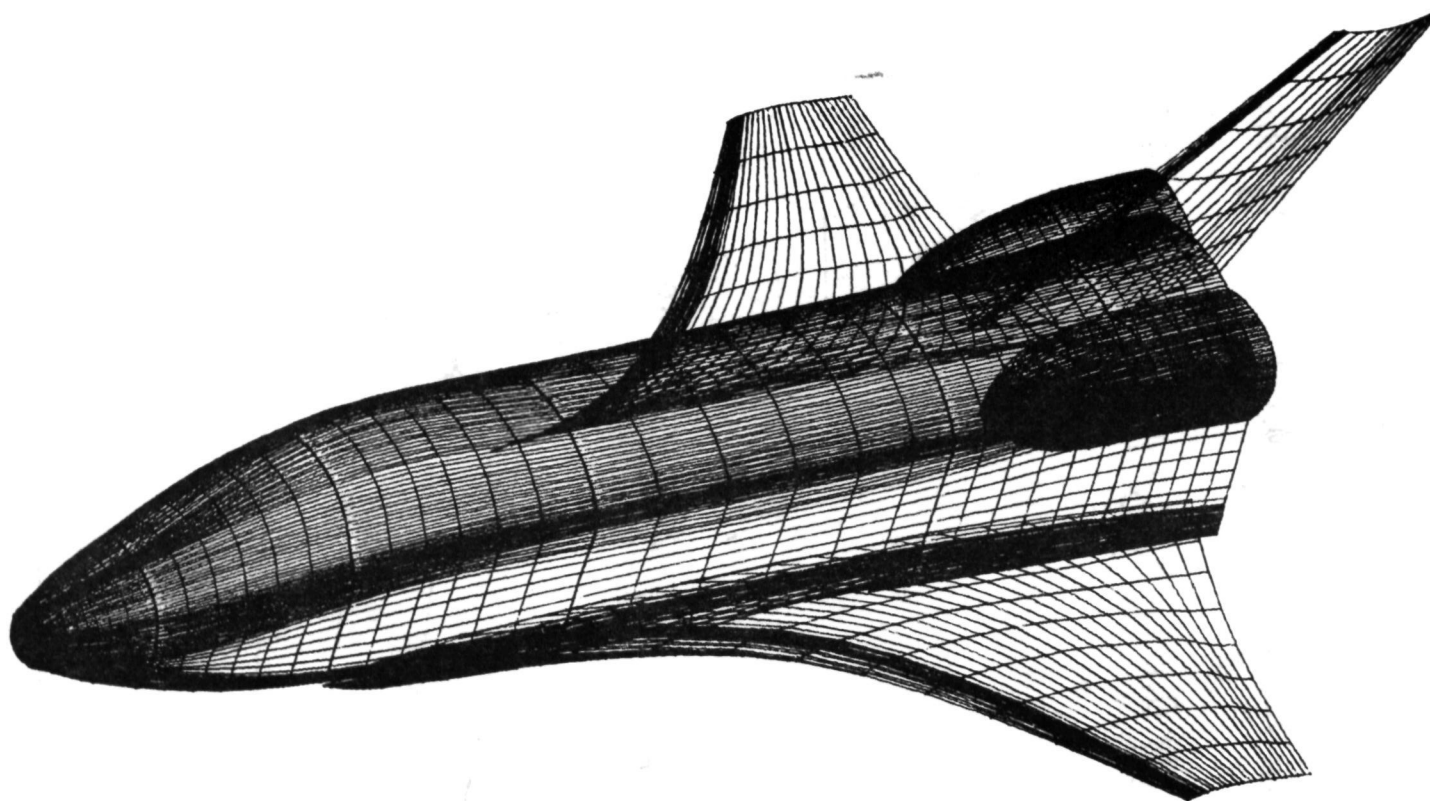


Figure 5. - Surface plot examples.

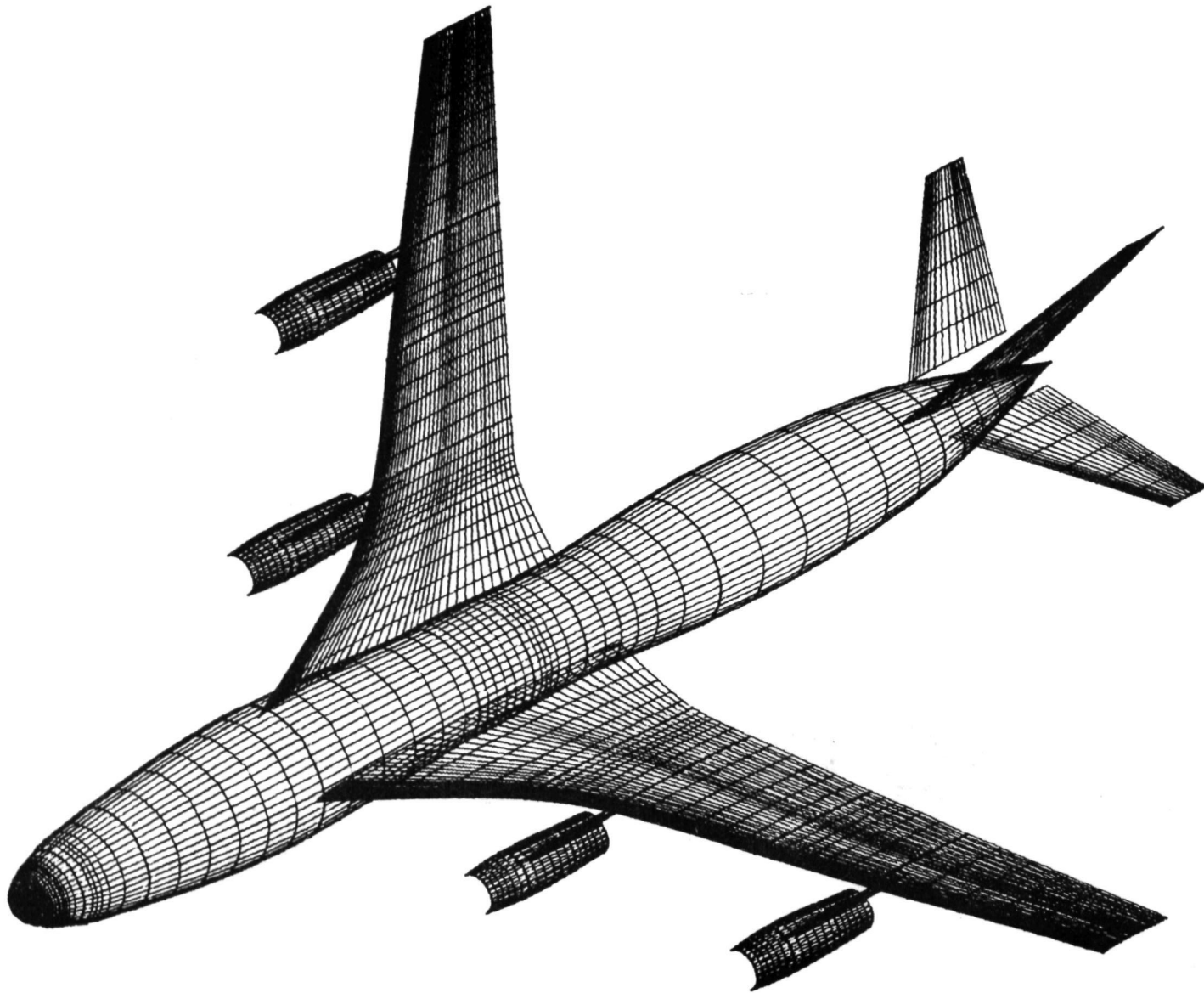


Figure 5. - Continued.

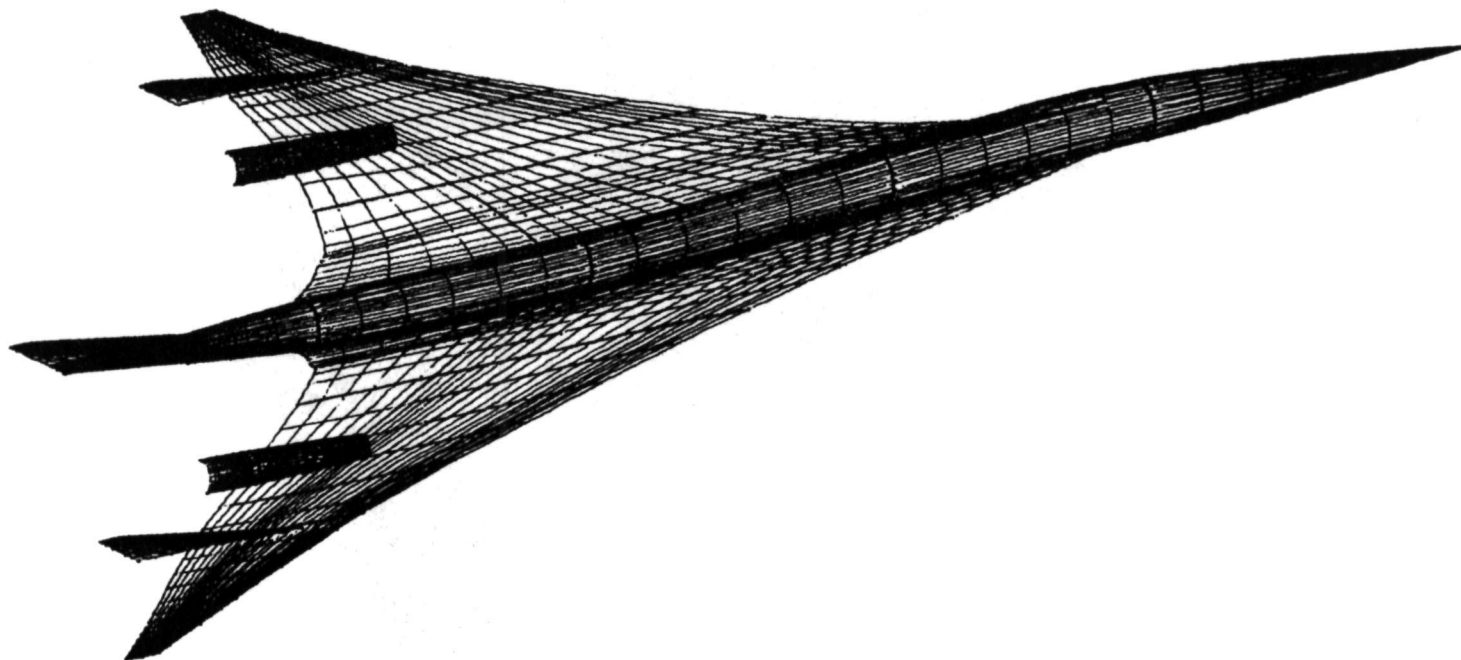


Figure 5. - Continued.

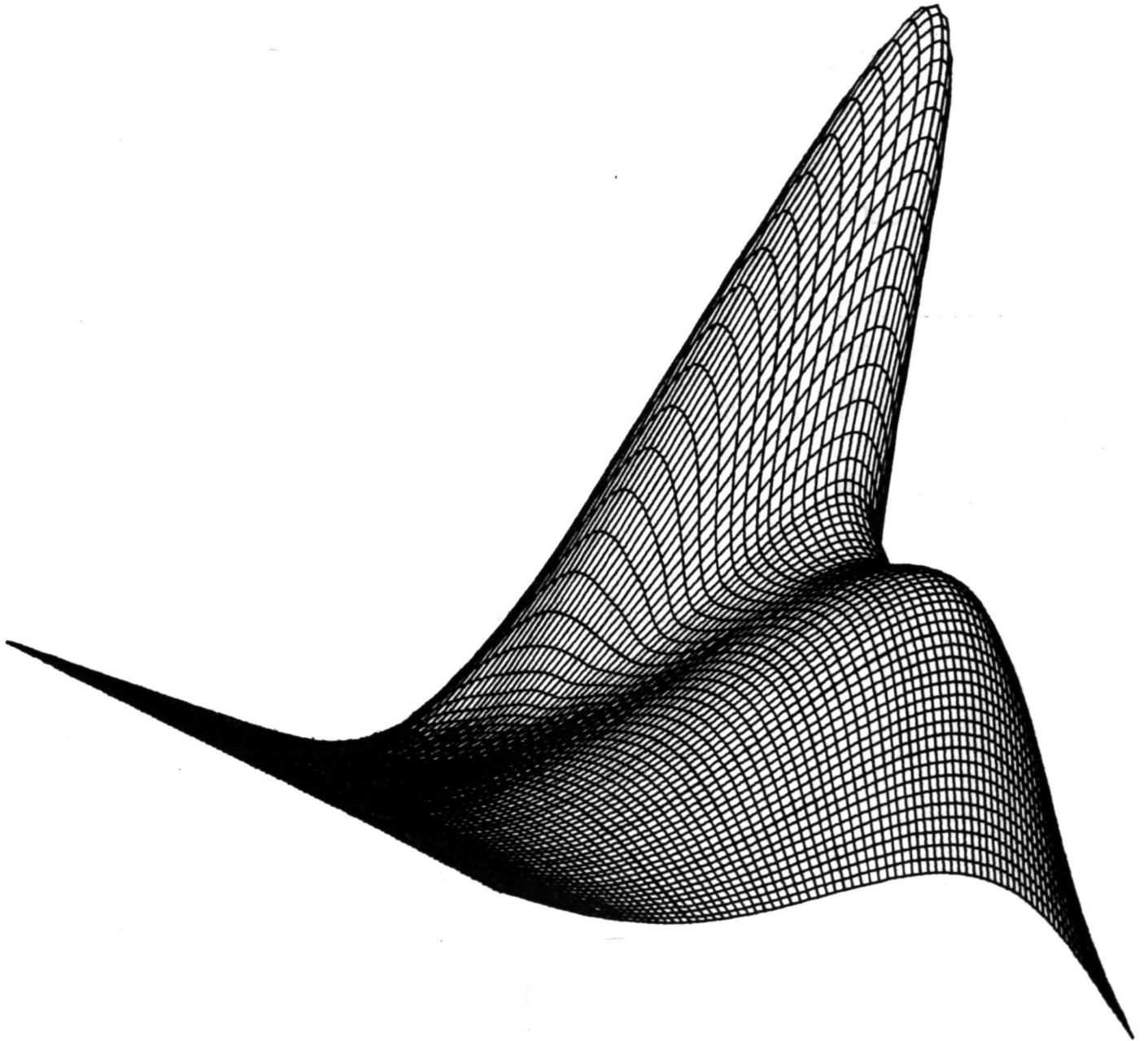


Figure 5. - Concluded.

Patch Equation:

$$\begin{array}{rcl} X(u, w) & = & U S_x W^t \\ V(u, w) = U S W^t & = & Y(u, w) = U S_y W^t \\ Z(u, w) & = & U S_z W^t \end{array}$$

Equation of a Plane :

$$a x + b y + c z - d = 0.$$

then:

$$U [\alpha S_x + b S_y + c S_z] W^t - d = 0.$$

with:

$$G = \alpha S_x + b S_y + c S_z$$

$$U G W^t = d = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} G \\ G \\ G \\ G \end{bmatrix} \begin{bmatrix} w^3 \\ w^2 \\ w \\ 1 \end{bmatrix}$$

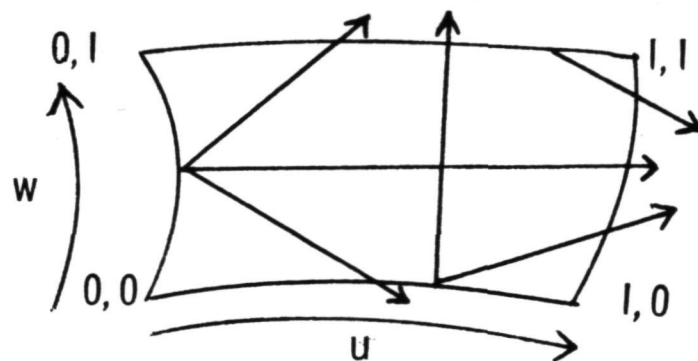


Figure 6. - Cross sections or contours through a patch.

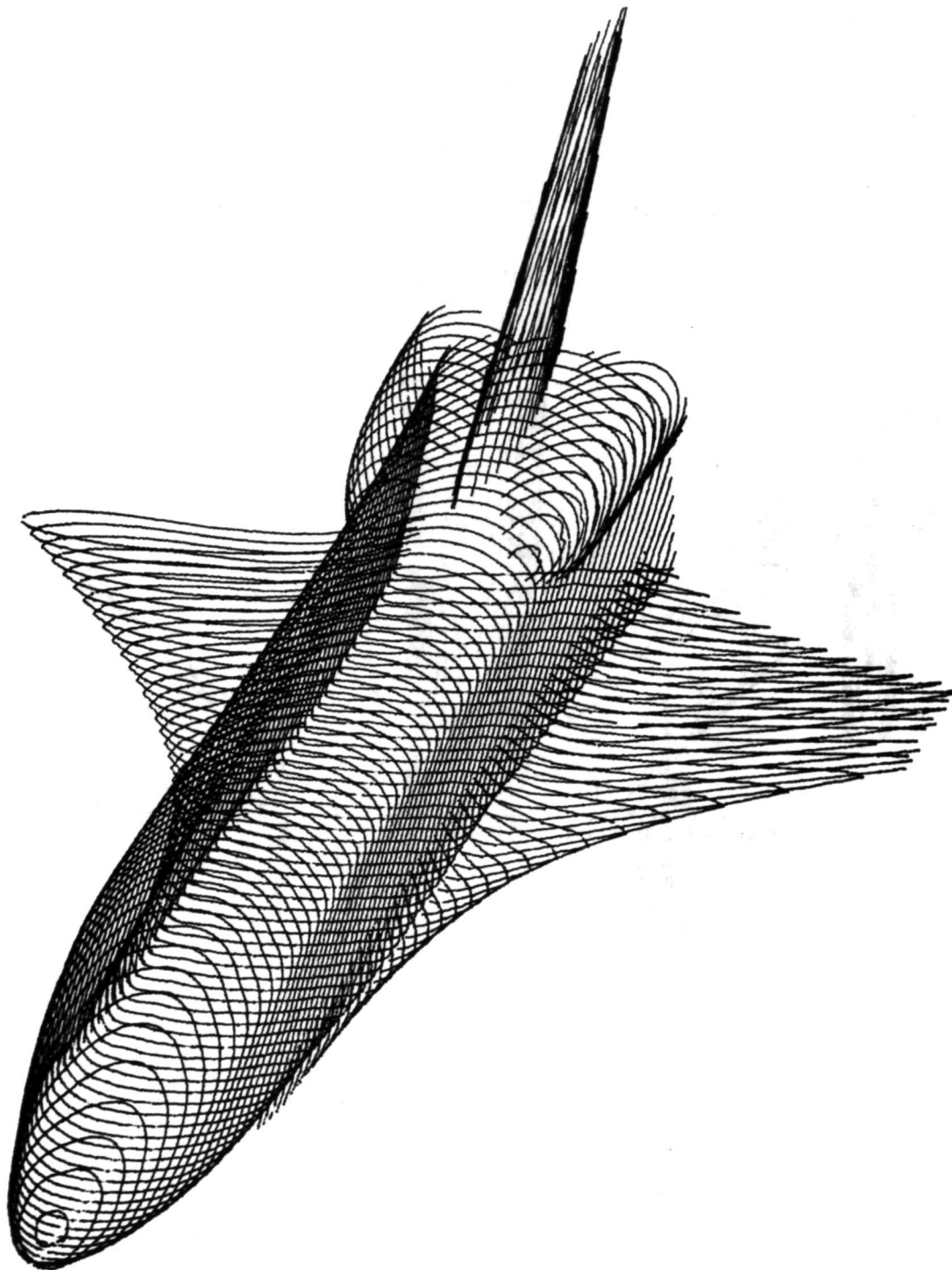


Figure 7. - Cross section plot examples.



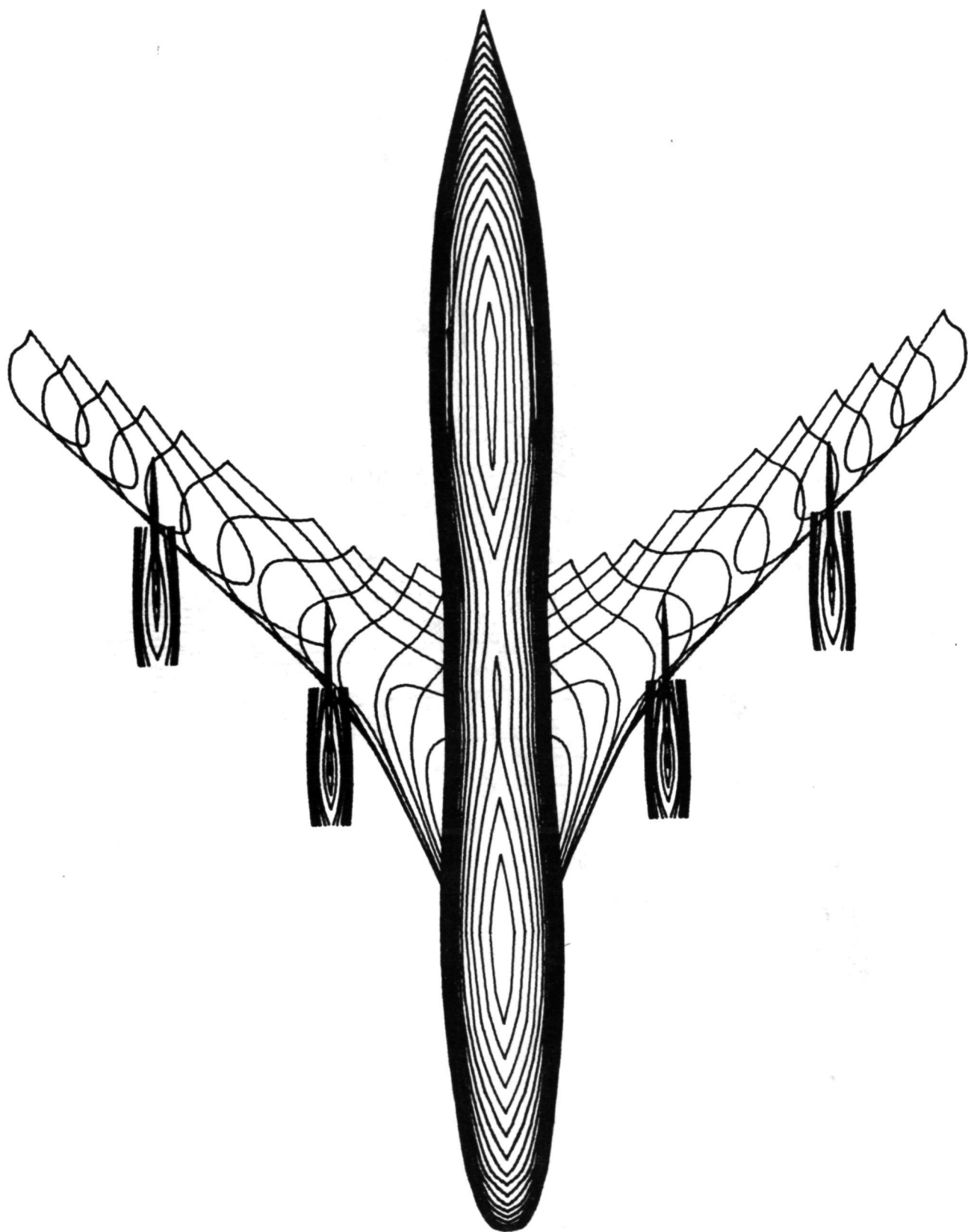


Figure 7. - Continued.

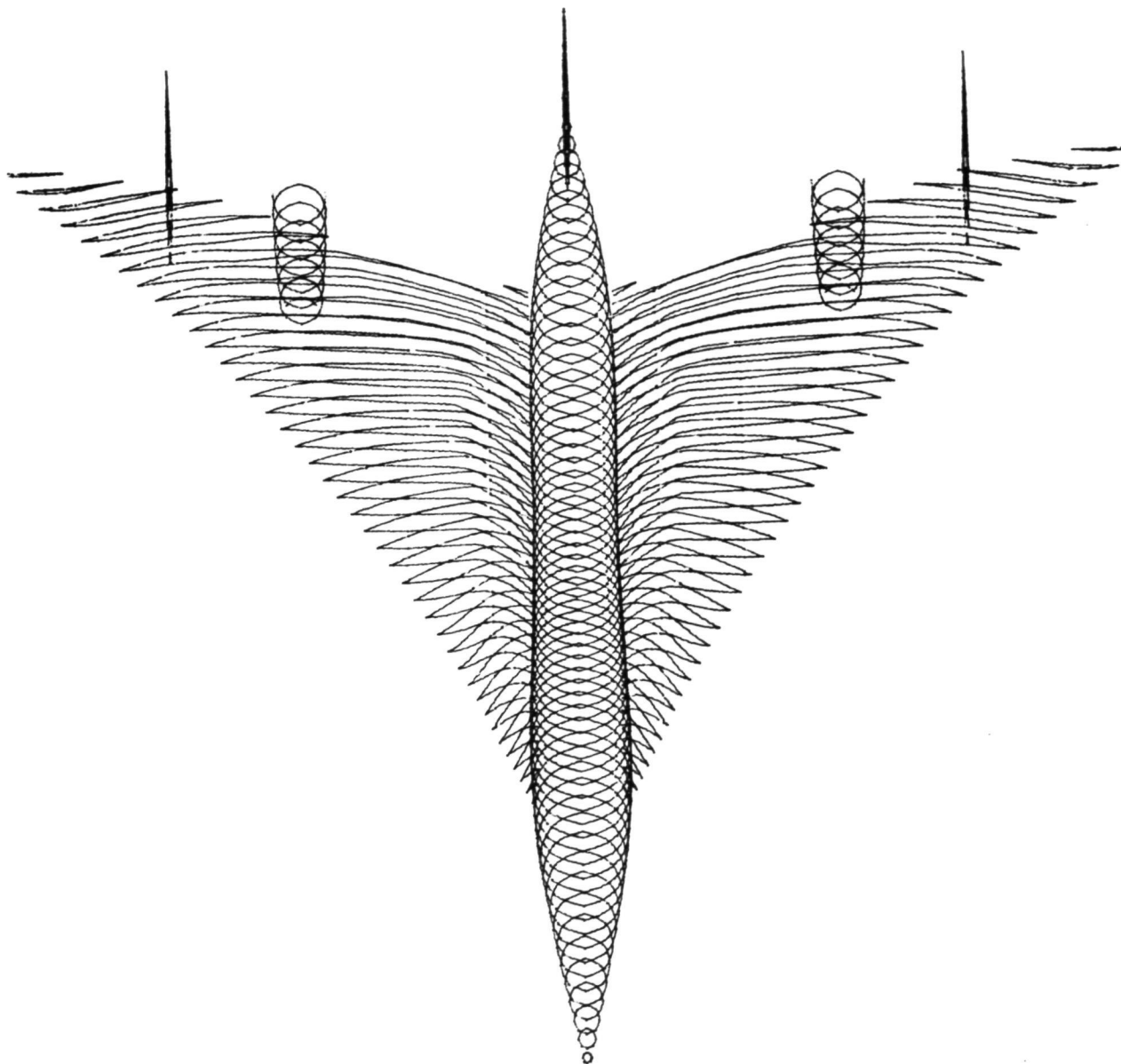


Figure 7. - Continued.

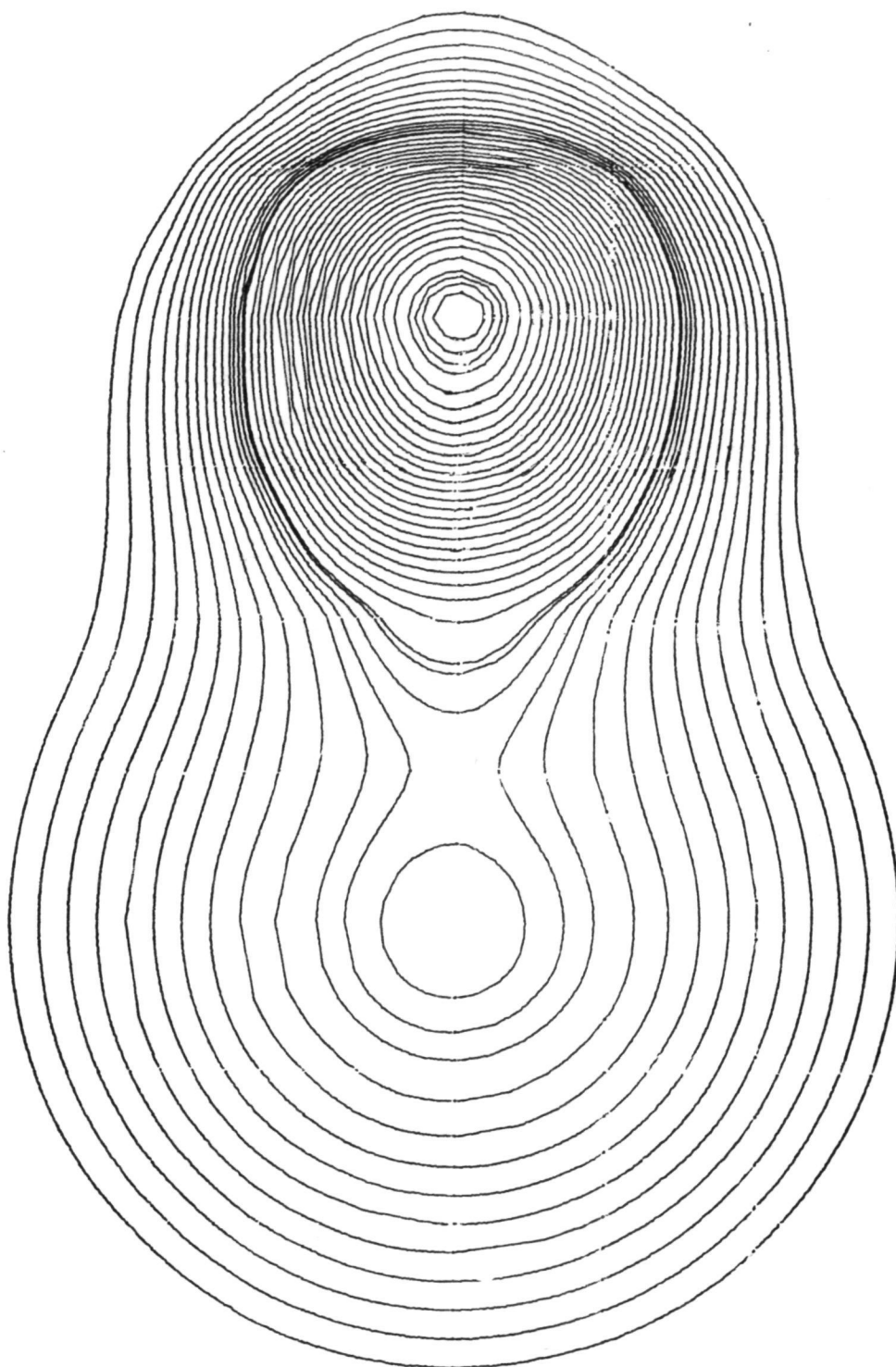


Figure 7. - Concluded.

# MESH GENERATION FOR TWO-DIMENSIONAL REGIONS USING THE TEKTRONIX

## DVST (DIRECT VIEW STORAGE TUBE) GRAPHICS TERMINAL

Verlan K. Gabrielson

Sandia Laboratories, Livermore

### ABSTRACT

This paper describes the code DVMESH and the use of the Tektronix DVST graphics terminal for applications of preparing mesh data for use in various two-dimensional axisymmetric finite element stress analysis and heat transfer codes.

### INTRODUCTION

The use of direct view storage tube (DVST) terminals, such as the Tektronix 4002A and 4014-1, for various graphics and interactive applications has expanded greatly during the past few years. This is due to their relatively low cost and their adaptability to the terminal networks used on most computer systems.

At Sandia Laboratories, Livermore (SLL), the DVST terminals are being distributed throughout the laboratory. These terminals are interfaced to a Control Data Corporation 6600 computer and are accessed via CDC's Intercom software system. The accessibility, ease of use, and adequate display area of the terminals make this graphic system effective for interactive applications which do not require large data bases or high rates of data transfer. (This does not imply that such applications cannot be implemented, but they increase the time between interactive responses. The time the user waits for a response is a primary consideration in any interactive application and becomes more sensitive in a system like SLL's since the data and programs are processed and stored on a host computer which is servicing a large number of users.)

To support this expansion of graphics facilities, a number of programs have been coded to evaluate applications for which interactive graphics can be used most effectively. The DVMESH code uses the DVST terminal to prepare mesh data for various finite element stress analysis and heat transfer codes. The project was designed to evaluate techniques, attributes, and limitations of the DVST terminal for this type of application.

The need is quite real; designing finite element models is normally one of the more time-consuming tasks of the structural engineer. The task of designing the proper mesh for an analysis is quite dependent on having an adequate pictorial display of the mesh. Thus design time depends on the speed with which graphic displays can be created, changed, and verified. For this reason, a mesh code designed around the capabilities of the terminal for problems requiring a lot of graphics verification should become a very useful tool for the structural engineer.

The design parameters and goals of the DVMESH code have been to:

- (a) Make it applicable to two-dimensional regions
- (b) Allow its input data to come from various sources
- (c) Implement available interactive features of the DVST, with special emphasis on data sets entered at the terminal keyboard
- (d) Design a standard output data set which will allow the mesh to be designed in modules
- (e) Design for approximately 25,000 words of memory
- (f) Provide necessary intermediate files in order to recover after abnormal or normal interruptions

The following sections describe the Tektronix terminals, details of DVMESH and its implementation on a sample problem, and some general comments concerning the DVST terminal for this type of application.

#### THE TEKTRONIX TERMINAL

The DVST terminals use the CDC Intercom software system which resides on the CDC 6600. Applications programs are written in FORTRAN, using system subroutines for displaying text, constructing line vectors, and initiating the cross-hair cursor. The display screen is a write-only display which can be refreshed only by erasing the entire screen.

The 4002A terminal being used has a display screen of 1024 (X) by 780 (Y) viewable points and space for thirty-nine lines of alphanumeric text. The terminal has a standard keyboard and an interactively driven cross-hair cursor. These features can be seen in Figure 1.

Features of the system which are attributes in graphics applications are:

- (a) An easy-to-use keyboard in which all entries are displayed in a scratch area on the screen for verification of the line of data as it is being keyed.

- (b) Hard copy prints of any display are easily obtainable.
- (c) Using the Intercom system as the interface between the terminal and the computer allows the user to communicate directly with the SCOPE operating system on the 6600. This provides access to the mass storage devices on the 6600 and permits the use of the UPDATE and file processing programs, plus the Intercom text editor program.

Features which are limitations in graphics applications are:

- (a) A minimum of interactive hardware is provided. All displays have to be designed by the program residing on the host computer. Such features as windowing, rotation, etc., are done by the user's program.
- (b) The system cannot permit a selective erase. Thus more effort is required in programming the displays for prompting messages, data verification, and modifications to subregions of the display.
- (c) In effective interactive processing the wait time between operations should be minimized. This time depends on the rate of data transfer and the size of the data buffer. With the Intercom system, it also becomes a function of the number of users on the network and the workload on the 6600.

The DVMESH code was designed and implemented on the above system, but recent deliveries of 4014-1 displays with their 4906 (X) by 3120 (Y) viewable points and up to sixty-four alphanumeric line capability will enhance the capabilities of the program. In addition, the PLOT-10 software package available from Tektronix will be implemented on the system.

#### THE DVMESH CODE

The DVMESH code applies to two-dimensional regions. These regions may have irregular boundaries, material interfaces, voids, etc. The ability to create an adequate mesh over the regions depends on the user's skill in using the functions available in DVMESH, and the restrictions imposed by the stress analysis or heat transfer code for which the data is being prepared. In all of these codes the meshing problem is similar to those described in the survey paper on mesh generation (reference 1).

A space is subdivided into quadrilateral elements consistent with the needs of the problem and the type of analysis. The mesh data required are the coordinates of the nodes defining the corners of the quadrilateral element. The DVMESH code is designed to produce mesh data for finite element codes which require that all the nodes be mapped onto a unit grid, such that each node can be identified by an integer pair (i, j) denoting specific rows and columns in the grid. Figures 2

and 3 illustrate this mapping. This procedure, which is common in many finite element codes and creates difficult problems in meshing irregular regions, defines the connectivity of the mesh and simplifies the problem of matrix storage and boundary definitions. For this type of application, in which data preparation and graphics verification are very time-consuming, the interactive capabilities of the DVMESH code have been quite useful.

DVMESH interactive capability is implemented in three phases: input boundary definition, mesh development, and output generation. The following sections give a brief description of each phase. The code uses only the hardware features of the keyboard and cross-hair cursors. "Functions" in the following descriptions refer to programmed routines in the DVMESH code which can be used by entering special keywords at the terminal.

### INPUT BOUNDARY DEFINITIONS

The input data set consists of coordinates of points, line segments, and circular arcs which define the basic boundaries of the regions to be meshed. An illustration of a sample data set is shown with the sample problem. Four input options have been implemented, providing considerable flexibility to the user for data preparation. The options include:

- (1) Data sets entered at the terminal
- (2) Punched card records stored on disk file
- (3) Data sets generated by digitizers
- (4) Data sets extracted from large data bases generated by the APT (automatic programming tool) processor used for automatic drafting purposes

The input data records are identified by line numbers and can be displayed and edited by entering functions at the terminal with the line identifiers. These functions, which are used in an interactive mode, include inserting, changing, and deleting specified lines in the input data set and displaying selected data sets.

A graphics display of the boundary data set can be obtained at any time during input; this provides a means to verify, correct errors, and make adjustments to the data at the terminal. The user iterates on these operations until the boundary data set is properly defined. Functions used in the interactive mode include plotting the data within given min-max values and "windowing" by contracting or expanding the plot about a point which is identified with the cross-hair cursors.

## MESH DEVELOPMENT

The meshing algorithm is quite simple. The mesh is designed as a rectangular array of quadrilateral elements having (m) rows and (n) columns. Each node of an element is uniquely identified by an integer pair (i, j) representing its row-column location in the grid, and the coordinates of the nodes are stored at equivalent locations in their respective arrays. The user defines the nodes at selected locations on the boundary data shown on the display by use of a special function and assigns to each its (i, j) location. The entire mesh can be defined by the user, but normally only a minimum set of nodes need be identified. All nodes not defined by the user are computed by linear interpolation along each row or column. This is illustrated in the sample problem.

## INTERACTIVE PROCEDURE

When the meshing code is initiated, the boundary data set which exists within the current min-max values is displayed and the maximum size of the mesh arrays for the given problem is entered at the terminal by the user. The user proceeds by defining nodes on the perimeter of the boundary data. When a sufficient number have been defined, a mesh is constructed within the defined nodes. By observing the mesh the user proceeds by selecting the necessary functions to accomplish various tasks. The result of each function is displayed and a mesh can be reconstructed at any step in the process. This interactive procedure is continued until a satisfactory mesh is developed over the given boundary data set.

## THE PICK FUNCTION

The basic meshing function is PICK, which is used to define node locations by use of the cross-hair cursor. This function, entered at the keyboard by the user, displays the cross-hair which can be interactively moved to any location on the display. The cursor is turned off by a keyboard interaction. When the cursor is turned off, the user records the node's location in the mesh by entering at the terminal its (i, j) coordinates. Following this entry, an X is displayed at the point selected by the program for this node and the cross-hair is reinitiated for further processing.

Interaction involving the cross-hair cursor requires the use of the system subroutine LOC. When the cursor is turned off, the function returns the coordinates of the cross-hair intersections and the keyboard entry used in the interrupt. This provides a valuable programming tool, since each key can represent a functional response. In this application



the keys N, P, and L provide alternatives in the location of the node at the cross-hair intersection. The N option locates the point at the cursor and the X option transfers control out of this procedure. The P and L options provide a means to locate the node on the boundary data set. P implies that the node is located at the point in the boundary data set nearest the cross-hair, and L implies locating the node on the nearest line segment.

## SPECIAL FUNCTIONS

After an initial mesh has been constructed from the nodes defined by the PICK function, further development of the mesh can be done by use of the following functions. These functions are illustrated in the sample problem.

- CIRP is a function for locating nodes on a circular arc by identifying three nodes on the arc which have been previously defined.
- BFIT is a function similar to PICK which moves each node along a given row or column to the nearest point in the boundary data set.
- VOID is a function which permits regions in the mesh to be defined as voids on subsequent displays.
- INIT is a function providing a means to reinitialize the entire grid or parts of it when errors have to be corrected or adjustments made.
- SHFT is a function which allows nodes of a given row or column to be located as normal translations from a defined row or column. This is very useful in obtaining accurate definitions in meshing structures composed of layered materials, interface gaps, etc.

Nodes not defined by the above functions can be generated by one of the grid functions:

- GRDI displays the mesh by interpolating between defined nodes on each row of the grid, then interpolating between defined nodes on each column.
- GRDJ similar to GRDI, except nodes are defined on each column, then on each row.
- GRIJ used for defining nodes within a section of the grid having convex or concave boundaries which may not mesh properly using GRDI or GRDJ. It requires that all nodes on the boundary of the section be specified prior to use.

## ERAS

In practice, several iterations will be required to obtain the desired mesh. This may require many erasures of the display. The ERAS function provides this capability. The display is erased and the boundary data set plus the status of the defined nodes are reset to the definitions prior to the erase. The defined nodes include all that have been defined by the above functions except those computed by interpolation in the GRDI and GRDJ functions.

## OUTPUT PREPARATION

The output data file prepared by the DVMESH code consists of problem identifiers and the coordinates of the nodes identified by a specific row and column. This array, stored on disk file, is the standard output file which can be implemented by the various analytical codes. It also provides a restart capability for DVMESH.

Interactive features at this level permit the mesh to be viewed in detail and allow location of a given node by (x, y) and (i, j) coordinates; this may be useful for boundary and material identifications in the finite element analysis.

With a memory allocation of 25,000 words, meshes of over 2,000 nodes can be developed by DVMESH. For the case of large problems, the code output file is designed so that the mesh can be developed in sections.

## SAMPLE PROBLEMS

The following sequence of figures illustrates how a mesh is developed using DVMESH at an interactive terminal. The procedure illustrates only one of many possible procedures to obtain a mesh.

Input data options:

```
TYPE INPUT FILE AND PROBLEM HEADER
60--TYPED INPUT
40--PUNCHED CARD INPUT
20--DIGITIZED DATA FILE
42--APT DATA FILE
13--RESTART DATA FILE
```

All displays shown in this section were prepared from hard copy prints of displays on the terminal. For example, input data entries from terminal proceed as follows:

```

60      SAMPLE PROBLEM
      TYPE DATA CARD
DATA 1 1 1
      TYPE DATA CARD
SIDE 1 13 0. 0. 2. 4.2 .2
      TYPE DATA CARD
SIDE 2 1 3 1. .5 2.5 .5 2.5 .7
      TYPE DATA CARD
SIDE 2 1 2 1.0 .7 1.0 .5
      TYPE DATA CARD
ARC 4 1 2 0. 4. 2.6 270. 340.
      TYPE DATA CARD
ARCP 6 1 2 1.9 1.6 2.4 2.15 2.8 3.1
      TYPE DATA CARD
ARCP 7 1 2 1.9 1.6 1.8 1.4 2.2 1.1
      TYPE DATA CARD
ARCP 7 1 1 2.2 1.1 3.0 1.15 4.2 1.3
      TYPE DATA CARD

```

Procedure is completed when END is typed.

Editor Phase options are PLOT-CHG-INS-DEL-LIST.

Display of data entered at terminal using LIST function:

```

1 DATA 1 1 1
2 SIDE 1 13 0. 0. 2. 4.2 .2
3 SIDE 2 1 3 1. .5 2.5 .5 2.5 .7
4 SIDE 2 1 2 1.0 .7 1.0 .5
5 ARC 4 1 2 0. 4. 2.6 270. 340.
6 ARCP 6 1 2 1.9 1.6 2.4 2.15 2.8 3.1
7 ARCP 7 1 2 1.9 1.6 1.8 1.4 2.2 1.1
8 ARCP 7 1 1 2.2 1.1 3.0 1.15 4.2 1.3
9 END
TYPE -PLOT-CHG-INS-DEL-LIST-

```

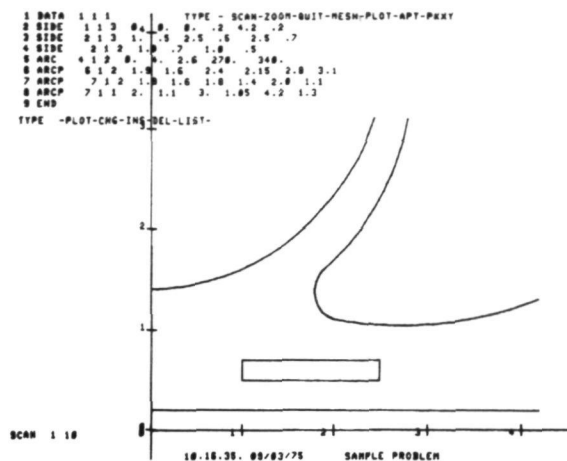
Example of changing data entry 2:

```

CHG 2
      TYPE DATA -- FOR INS USE LAST TO END
SIDE 1 1 3 0. 0. 0. .2 4.2 .2
TYPE -PLOT-CHG-INS-DEL-LIST-

```

The entry PLOT displays the data graphically:



Plotting phase functions available at this phase of problem:

TYPE - SCAN-ZOOM-QUIT-MESH-PLOT-APT-PKXY

Transfer is made to the meshing program when the MESH function is entered.

Before meshing proceeds, the user must have some insight into the size of grid desired and relative locations of the maximum and minimum (i, j) node identifiers. This can be done by drawing a rough sketch of the desired mesh as a guide to the interactive development.

The maximum row and column is entered when the MESH function is typed. All references to node identifiers in the mesh code must be within these bounds.

The following illustrates the available functions for mesh development:

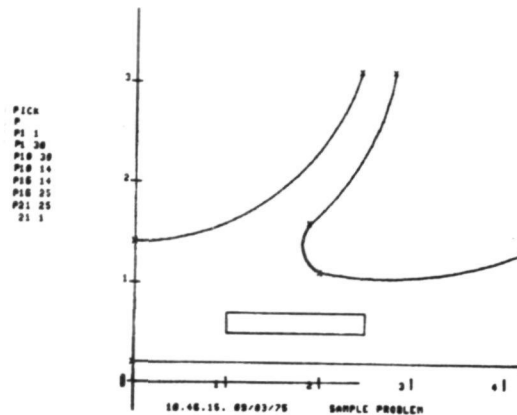
```

MESH SIZE 21 30
157
PICK
TYPE--PICK-ERAS-GRDJ-GRDI-PLOT-BFIT-SAVE-INIT-VOID-
SMFT-EXIT-LOCP-PKXY-FITQ-GOBB-CIRP-GDIJ
TYP

```

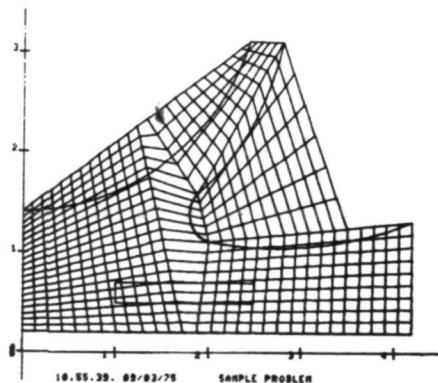
This example illustrates the use of some of these functions for developing a mesh over the above boundary data set. The first function used is PICK in which the cross-hair cursor is moved to the lower-left corner of the boundary data. When P is typed the node is located at the end point of the line segment and identified by entering the (i, j) values of (1, 1).

The interaction continues by use of the cross-hair and the PICK function until a set of nodes is defined on the perimeter of the boundary data set. For this case the PICK entries are defined in a counterclockwise direction. The following displays these nodes:



The nodes defined above are a sufficient set for constructing a mesh over the region.

The following display results when GRDI is entered.



At this point the functions BFIT and CIRP are used to locate given rows and columns on the boundary data set. In this example row 21 is located on the circular arc passing through nodes (21, 1), (21, 10), and (21, 25). The node (21, 10) is located on the circular arc by the BFIT or PICK function. From this data the nodes (21, 2), ..., (21, 24) are defined on the circular arc in equal angular segments by the function entries:

```
BFIT 21 10
CIRP 21 1 21 10 21 25 1
```

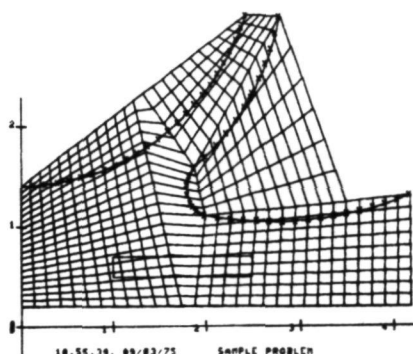
In like manner node points between (16, 14) - (16, 25), (10, 14) - (10, 30), and (10, 14) - (16, 14) are located on their respective circular arcs:

```

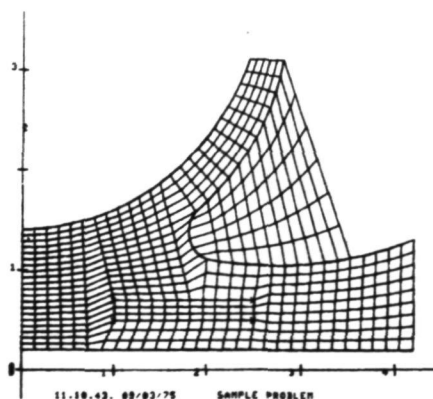
BFIT   16  20
CIRP  16 14  16 20 16 25  1
BFIT   10  20
CIRP  10 14  10 20  10 30  1
BFIT   12  14
CIRP  10 14  12 14  16 14 -1

```

The X's on the boundary sets in the figure illustrate how the functions are verified on the display.



To redisplay the mesh, the ERAS function is used followed by GRDI, which interpolates new values for all nodes not defined by the special functions. The following display results:

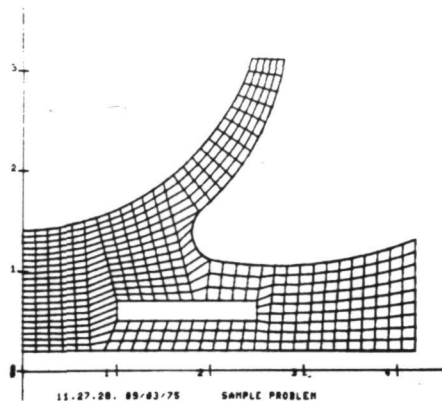


This display also shows the result of defining four nodes about the rectangular region using the PICK function.

To remove nodes which lie in regions defined as voids the following entries are made:

```
VOID 11 15 15 25
VOID 6 8 7 17
```

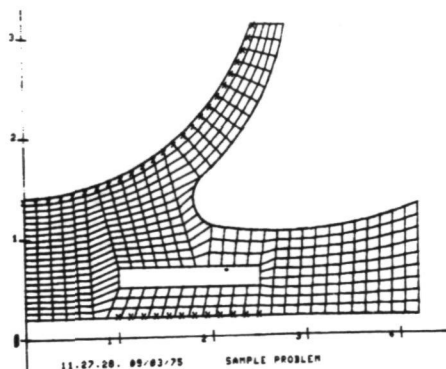
These nodes are verified on the display, then ERAS and GRDI functions are entered, resulting in the following mesh.



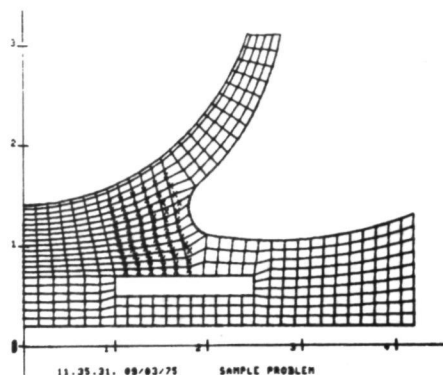
To illustrate the use of the SHFT function the figure shows where nodes are located when the following entries are made:

```
SHFT 21 1 21 25 .84 -1 0
SHFT 4 7 4 18 0. -.3 -3 0
```

The first relocates row 20 a .04 normal distance from row 21. The second entry relocates selected columns of row 1 a fixed distance from row 4.

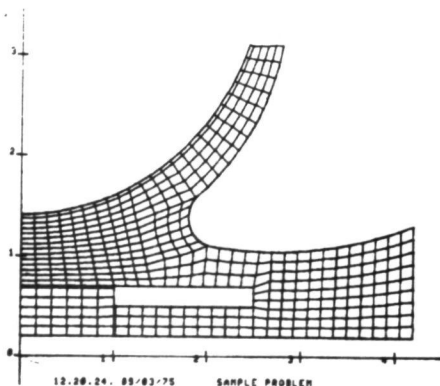


To reset the nodes in the convex region of column 14 the GDIJ function is used in the region bounded by the nodes (7, 7) and (20, 14) as shown in the display:



The final mesh in the following display shows the result of GDIJ function plus the use of the SHFT function for defining the interfaces along sections of row 7 and column 7. The SHFT function entries were:

```
SHFT 1 1 1 7 10 148 5 0
SHFT 1 7 4 7 102 .0 0 1
```



The data sets are now processed onto an output file which can be used in various analytical codes. As can be observed, the procedure to obtain a mesh is not fixed but is dependent on the user, his needs, and his experience. The goal is to initiate a crude mesh over the region and then refine the mesh by visual observation and the use of the various special functions.



## GENERAL OBSERVATIONS

The development of DVMESH provided considerable experience for interactive meshing applications. Using a system with a minimum of interactive features and relatively slow response time required numerous tradeoffs between the frequency of refreshing the display and the complexity of the task for each function.

Implementing DVMESH as an interactive process implies that the user will develop the grid visually, since it is quite inefficient to consult notes, procedures, etc. while at the terminal. This requires instructions and prompts to be displayed, as well as verification of functions implemented by the user. These procedures are normally used in interactive programs but are essential in this application, since the response of the system is quite variable. These additional displays have the effect of filling the screen quickly, since no selective erase (a feature of larger graphic systems) is available.

The design of interactive programs for a DVST device requires some special considerations associated with the erase function. Since an erase results in a total erasure of the display, special efforts must be implemented to reproduce meaningful data with minimum effect on time and prior efforts. As shown in the sample problem, a number of erasures were required to complete a problem. To minimize the effect of the erase, a file is created which contains node definitions generated by the functions PICK-CIRP-BFIT-VOID-INIT-SHFT, etc., prior to the erase. Processing this file after the erase resets the meshing procedure to a point prior to the last grid instruction from which processing can continue. This requires that after each erase the graphics display is rebuilt, which can be time-consuming. To reduce this time requires special programming efforts such as minimizing axis annotation, design of mesh, etc.

Another important item of any interactive program is recovery when erroneous data is entered at the terminal. In DVST applications the programs must be coded to check on data consistency, proper data values, number of entries, illegal characters, etc. The programming time spent on this feature is worth the effort, since transmitting erroneous data is easily done at the terminal. In DVMESH a very useful subroutine called MASK has been used for this purpose. It provides a way to test for data consistency and the proper number of data entries for each type of function. It allows all data entries to be entered in a free field format, which is an essential feature for DVST terminal inputs.

An aspect of the DVMESH code which has proven to be quite valuable is the ease of making modifications to existing meshes. Since most of the meshes apply to design problems, the need to make parameter studies by making minor adjustments to an existing mesh occurs quite often. With DVMESH, the existing data can be retrieved from the computer data files, adapted to the new design, and verified graphically in much less time than by previous procedures.

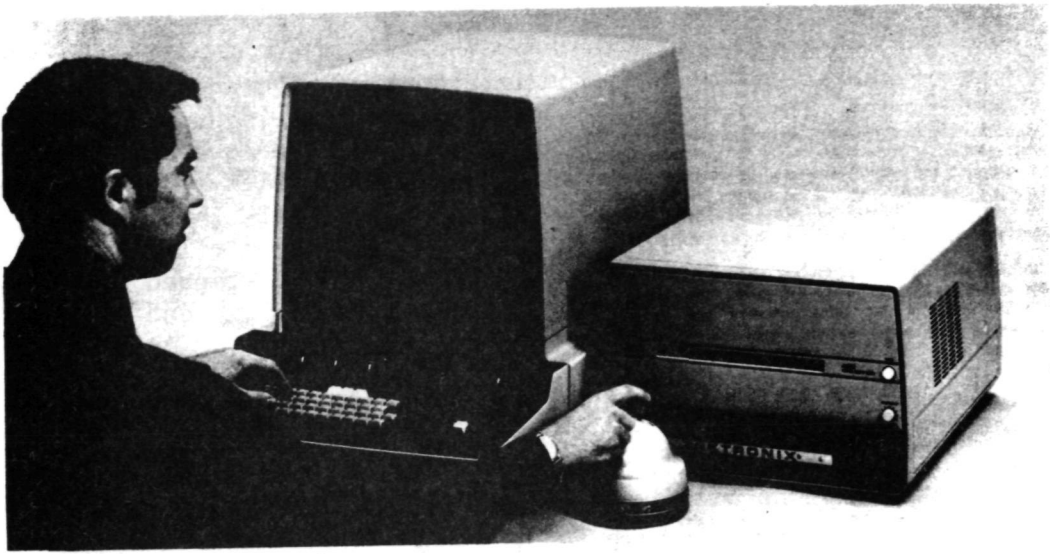
## CONCLUSION

Most of the goals noted in the introduction have been achieved in the DVMESH code. The code is in operation and is being used as a practical alternative at this time for constructing meshes over two-dimensional regions having irregular boundaries. The code provides a way for a user to design a mesh from a low-cost terminal by a visual, interactive procedure. The time spent preparing the mesh may be similar to conventional batch procedures for a given problem, but the elapsed time from start to a verified mesh may be much less since the mesh can be verified graphically at each step of the process.

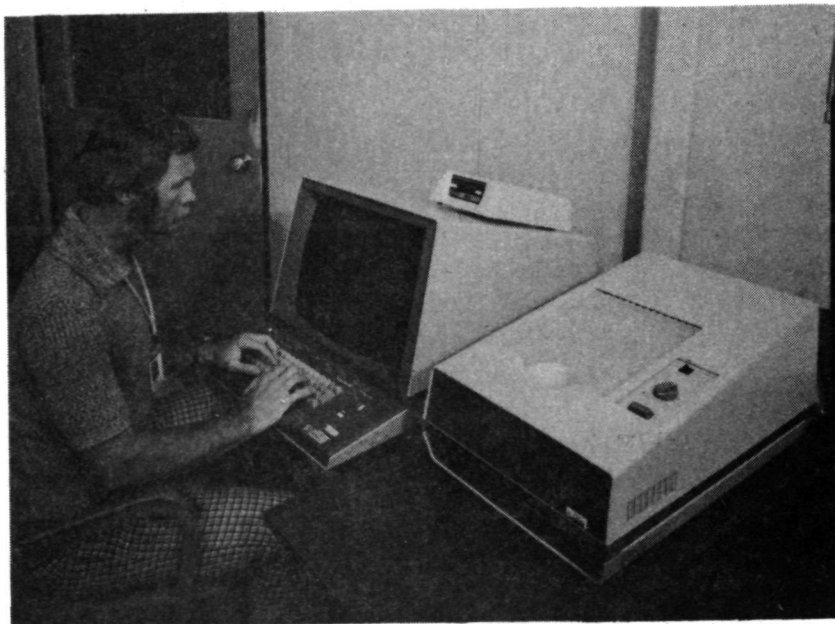
The code is written in FORTRAN but requires several system routines dependent on the graphics hardware used. It is available from the author on request. We anticipate that the design of the code will not be changed, but additions will be made with experience and new applications.

## REFERENCE

1. Buell, W. and Bush, B.: Mesh Generation - A Survey. Transactions of the ACME, Journal of Engineering for Industry, Feb. 1973.



(a) 4002A.



(b) 4014-1.

Figure 1.- Tektronix terminal facilities.

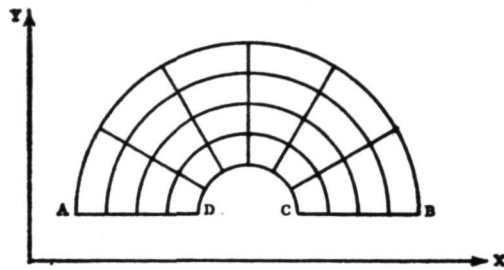


Figure 2.- Grid with equally spaced boundary points.

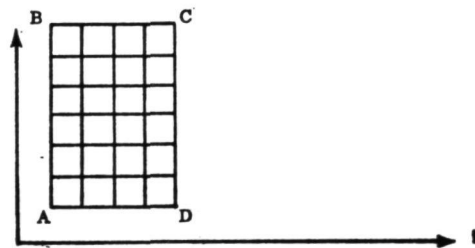


Figure 3.- I-J grid mapping from grid in figure 2.

Page Intentionally Left Blank

AN INTERACTIVE GRAPHICS PACKAGE FOR THE  
AUTOMATIC NODE RENUMBERING OF FINITE ELEMENT MATRICES\*

Ronald F. Boisvert and William G. Poole, Jr.

College of William and Mary  
and  
Institute for Computer Applications in Science and Engineering (ICASE)

SUMMARY

An interactive graphics software package which allows users to display the non-zero structure of large sparse symmetric matrices is described and methods used to implement it as a portable FORTRAN callable subroutine are summarized. In particular, the system permits the display of the resulting matrix after reordering the rows and columns, with the reordering scheme either defined by the user or automatically generated by the program with the aim of reducing matrix bandwidth and profile. Although the primary application of the package has been to the finite element analysis of structures, it is equally well suited to the many other areas of engineering and science which use sparse matrices.

INTRODUCTION

Frequently, in many areas of application, we must solve the linear algebraic system of equations represented by

$$Ax = b \quad (1)$$

where  $A$  is a non-singular  $n \times n$  symmetric matrix and  $x$  and  $b$  are  $n$ -vectors. Here we assume that  $n$  is moderately large (from about one hundred to several thousand) and that the matrix  $A$  is sparse; that is, the number of non-zero elements in the matrix is small compared to  $n^2$ .

In order to describe the non-zero structure of sparse matrices the concepts of bandwidth and profile are helpful. The bandwidth of a matrix  $A$  is defined as  $b = \max_{a_{ij} \neq 0} |i-j|$ , which is simply the radius of the smallest band about the diagonal which includes all non-zero components of the matrix. The

---

\*This paper is a result of work performed, in part, under NASA Grant NGR 47-102-001 while in residence at ICASE, NASA Langley Research Center. The work of the second author also was partially supported by the Office of Naval Research Contract N00014-73-A-0374-0001, NR 044-459. The first author is currently at the Computer Science Department, Purdue University.

profile is defined as  $p = \sum_{i=1}^n d_i$ , where  $d_i = i - \min_j \{j : a_{ij} \neq 0\}$ , that is, the sum of the distances from the main diagonal of the leftmost non-zero component in each row. The profile is exactly the number of non-zeroes in the lower triangular factor of the decomposition of  $A$  whose calculation is typically the first step in solving (1) by Gaussian elimination.

The non-zero structure of large sparse matrices is often used to reduce the otherwise restrictive storage and computational requirements for solving the linear system. Storage schemes for sparse matrices abound (references 1 and 2) with the band scheme being among the simplest to use. In this case, when the bandwidth of  $A$  is small, we can eliminate most zero components by simply storing only the diagonal bands of  $A$ . Other techniques incur even larger savings by taking advantage of the profile of the matrix. The non-zero structure of the matrix also greatly influences the computation time required to solve (1). The time required to perform Gaussian elimination on a full matrix is proportional to  $n^3$  while methods for band matrices typically require computation times proportional to  $nb^2$ . Once again, this can be reduced further by exploiting profile.

It is clear that for matrices with small bandwidth and profile considerable savings can be realized and, in fact, many codes are currently available for band or profile decomposition. Thus, the engineer wishing to solve (1) would normally pay particular attention to the non-zero structure of the matrix and, whenever possible, would seek that representation of the matrix which minimizes bandwidth and profile. Although band and profile schemes are not always the best way to treat sparse matrices, many of the matrices encountered in real applications, in particular, finite element approximations in structural engineering, are quite appropriate for the use of band or profile schemes.

The sparse matrix graphics package, by allowing the display of a representation of the non-zero structure of the matrix, gives the user a unique visual aid in the evaluation of various methods of storage and solution of the particular problem in question. In addition, by providing a matrix bandwidth and profile reducer as part of the package, the user may visually evaluate the worth of a renumbering of the rows and columns of the matrix which reduces bandwidth and profile to nearly minimal levels.

An example of the use of the graphics program is given in the following section with a description of the full capabilities and implementation of the graphics system following that. In the last two sections the algorithm used for matrix bandwidth and profile reduction is outlined and the results of the author's experience with the program are discussed.

#### EXAMPLE OF USE

The following scenario describes a typical use of the system. Suppose a design engineer, seated at a graphics terminal, is using the finite element

method for analysing some structure. After generating the elements, the designer numbers the nodes in some order, often an order which is easy to describe, and then generates a table defining the location of the non-zero components in the associated sparse matrix which represents properties of the structure being analysed. Because of interest in the effect of the nodal numbering on the non-zero structure of the matrix, the designer now invokes the sparse matrix graphics program via a subroutine call in the control program, causing the information at the top of Figure 1 to be displayed at the terminal. The design engineer then continues with the interaction shown in Figure 1, informing the program that the first display should be that of the input matrix shown in Figure 2.

Somewhat dissatisfied with his/her own nodal numbering, the designer might then request the program to generate a new numbering which reduces bandwidth and profile, thus producing the display of Figure 3. As a final comparison of the two numberings, the engineer may finally request the lower triangular halves of both matrices to be displayed at the same time, generating the display of Figure 4.

Not only does the design engineer get a nodal numbering which produces a small bandwidth and profile, but he also now has a visual conception of how effective the original numbering was.

#### IMPLEMENTATION OF THE DISPLAY SYSTEM

The sparse matrix graphics program is a highly portable FORTRAN program, requiring minimal local computing facilities. As demonstrated in the previous section, the graphics program has the following capabilities:

- \* Display the original matrix.
- \* Generate a reordering of the rows and columns which reduces bandwidth and profile, displaying the resultant matrix.
- \* Display the matrix after reordering the rows and columns according to a scheme input by the user.
- \* Compare any two of the above, displaying them at the same time on the screen.
- \* Matrices of any order may be displayed regardless of screen size.
- \* The size of the matrix display may be varied by the user.
- \* Either the full matrix or the lower triangular half may be displayed.

The package is invoked by a call to the subroutine SPARSE, with the generated renumbering returned as a parameter if calculated during the interactive session. In this way the package is easily interfaced with existing FORTRAN



coded interactive design systems.

The data structure chosen to represent the sparse matrix is the connection table, one frequently used in bandwidth and profile schemes (reference 3). The table has  $n$  rows and  $m$  columns, where  $n$  is the number of rows in the actual sparse matrix and  $m$  is the maximum number of off-diagonal non-zero components to be found in any row. The  $i$ th row of the table contains the column indices of all off-diagonal non-zero components in the  $i$ th row of the actual matrix. These column indices may be in any order. The values of the non-zero components themselves are never needed, only their indices. The renumbering schemes are represented as permutation vectors, whose entries show the order of the old rows and columns in the new matrix. For example, if the reordering vector is  $p$ , then the  $i$ th row in the original matrix becomes the  $p(i)$ -th row in the reordered matrix.

Due to the simplicity of the displays generated, rather minimal local graphics software capabilities are required to support the system. Indeed, any computation facility supporting interactive graphics terminals which provides FORTRAN callable routines to display a symbol and move the beam invisibly can easily implement the package on their system. These capabilities are interfaced with the sparse matrix graphics program via two user supplied routines, one which plots a row of user chosen symbols representing non-zero components at coordinates which are given, and one which simply moves the beam invisibly to a given location.

Displaying large matrices on most screens is a problem; for instance, on a terminal with a resolution of 1024 points by 780 points, a 1000 x 1000 matrix could not be displayed, even if each element were represented by a single point. To overcome this, the graphics program instead displays a related partitioned matrix; specifically, each symbol displayed represents an  $r \times r$  block of elements of the original matrix in which a non-zero occurs. The blocking factor  $r$  is chosen by the program and depends on the order of the original matrix and the display size requested by the user. The resulting display thus yields a visual description of the placement of non-zeroes, even when the actual matrix could not be displayed.

In order for the program to generate the coordinates of the symbols to be displayed, the calling program provides information concerning the dimensions of the screen, the size of the square symbol which the user has chosen to represent non-zeroes, and the size of the displayed characters resulting from ordinary FORTRAN write statements. The program then supplies display coordinates to the user-supplied graphics routines in the same units as the user implicitly defines the screen dimension information.

To run the program, some  $7n+2m$  core storage locations are required, in addition to the storage for the connection table ( $nm$  locations) and for the optional user supplied permutation ( $n$  locations). Thus the solution of large problems may be restricted by the core memory size. The largest part of these storage requirements is needed to implement the bandwidth and profile reduction subroutine, REDUCE.

## REDUCING BANDWIDTH AND PROFILE

The algorithm for bandwidth and profile reduction made available by the sparse matrix graphics program is one recently proposed by Gibbs, Poole, and Stockmeyer (reference 4). The actual FORTRAN implementation used, REDUCE, is detailed in reference 5. REDUCE has been found to be considerably faster than all other reduction codes in widespread use, generally superior for bandwidth reduction, and generally as successful as any other for profile reduction.

The algorithm can best be described in terms of the related adjacency graph,  $G$ , which has the property that there is an edge in  $G$  between vertices  $v_i$  and  $v_j$  if and only if  $a_{ij} \neq 0$  and  $i \neq j$ .

Step 1: Find the endpoints of a pseudo-diameter of the graph, that is, a pair of vertices that are at nearly maximal distance apart. This is done by a finite, iterative process of determining a vertex that is a maximum distance away from a given vertex.

Step 2: Given the pseudo-diameter endpoints  $u$  and  $v$  of distance  $k$  apart, partition the set of vertices into levels  $L_1, L_2, \dots, L_k$  such that adjacent vertices in  $G$  are in the same or adjacent levels and such that  $\max_i |L_i|$  is nearly minimized.

Step 3: Number the vertices of  $G$ , level by level, beginning at an endpoint of the pseudo-diameter.

A detailed description of the algorithm may be found in reference 4.

## EXPERIENCE

The sparse matrix graphics program was developed on a PRIME 300, a multi-programmed minicomputer with 64K of core storage at ICASE, NASA Langley Research Center (LRC). The program is interfaced with the University of Michigan Graph-Pack Library and has been run using primarily a Tektronix 4014-1 graphics terminal. The program is also running on the CDC 6600 at LRC under the KRONOS time-sharing operating system, again utilizing the Michigan graphics library, using a Tektronix model 4015-1 display terminal. Earlier batch versions of the program were run on an IBM 360/50 computer with a Calcomp model 765 plotter as well as on a CDC 6400 system with both Calcomp and Varian type plotters.

The package has been tested using most of the finite element matrices used to test subroutine REDUCE in reference 4, as well as on matrices generated in finite element work at LRC. In general, response times for the generation of displays (especially on the minicomputer system) were more than adequate for an interactive design environment. The block matrix scheme for displaying otherwise unrepresentable matrices was found to be most acceptable.

Although very large sparse matrices do not in general appear as sparse under this scheme, a good representation of the distribution of non-zero elements in the matrices is nevertheless provided.

#### REFERENCES

1. Rose, Donald J. and Willoughby, Ralph A. (editors): Sparse Matrices and Their Applications. Plenum Press, New York, 1972.
2. Tewarson, Reginald P.: Sparse Matrices. Academic Press, New York, 1973.
3. Everstine, G. C.: The BANDIT Computer Program for the Reduction of Matrix Bandwidth for NASTRAN. NSRDC Report 3827, 1972.
4. Gibbs, N. E., Poole, W. G., Jr. and Stockmeyer, P. K.: An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix. ICASE Report, July 22, 1974. To appear in SIAM Journal on Numerical Analysis, volume 13, 1976.
5. Crane, H. L., Gibbs, N. E., Poole, W. G., Jr. and Stockmeyer, P. K.: Matrix Bandwidth and Profile Reduction. ICASE Report 75-9, April 14, 1975. To appear in ACM Transactions on Mathematical Software, volume 2, 1976.

-----  
SPARSE MATRIX GRAPHICS PROGRAM \* VERSION 1.0 \* 10/75  
-----

CHARACTERISTICS OF INPUT MATRIX

ORDER = 310  
MAX NUMBER OFF-DIAGONAL ELEMENTS IN ANY ROW = 10  
BANDWIDTH = 302 PROFILE = 23357

(FOR ALL YES-NO QUESTIONS REPLY 1 FOR YES, 0 FOR NO)

WOULD YOU LIKE THE ORIGINAL MATRIX DISPLAYED ?

1

/ WOULD YOU LIKE A PERMUTED MATRIX DISPLAYED ON THE SAME SCREEN ?

0

/ WHAT SIZE BOX WOULD YOU LIKE THE PICTURE FIT INTO ?  
/ MAXIMUM SIZE IS 0.6640E 03

.664E3

/ ACTUAL DISPLAY MATRIX SIZE IS 0.5167E 03  
/ DISPLAY MATRIX PARTITIONED INTO BLOCKS OF SIZE 3

/ IF YOU WOULD LIKE ONLY THE LOWER TRIANGULAR HALF DISPLAYED ENTER 0.  
/ OTHERWISE ENTER 1

1

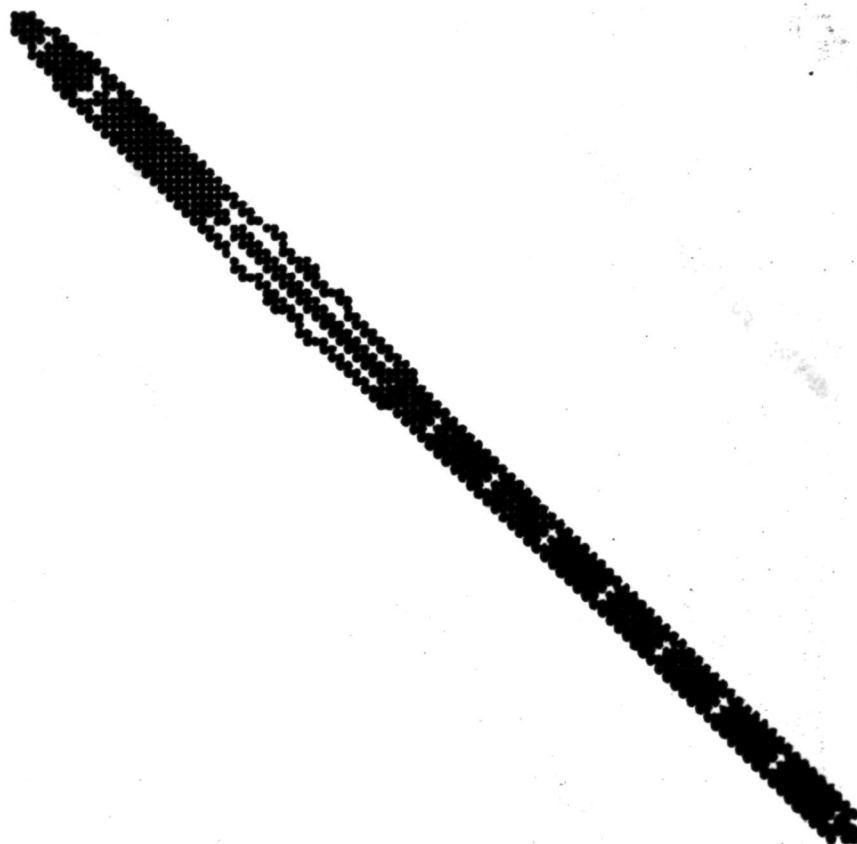
/ TO BEGIN DISPLAY, HIT ANY KEY, CLEAR SCREEN, AND RETURN

Figure 1.- Typical interaction with sparse matrix graphics program.



ORIGINAL MATRIX  
BANDWIDTH = 302  
PROFILE = 23357

Figure 2.- Display of input matrix.



GPS PERMITTED MATRIX  
SQUARED - 14  
PROFILE - 8706

Figure 3.- Display of matrix after renumbering.

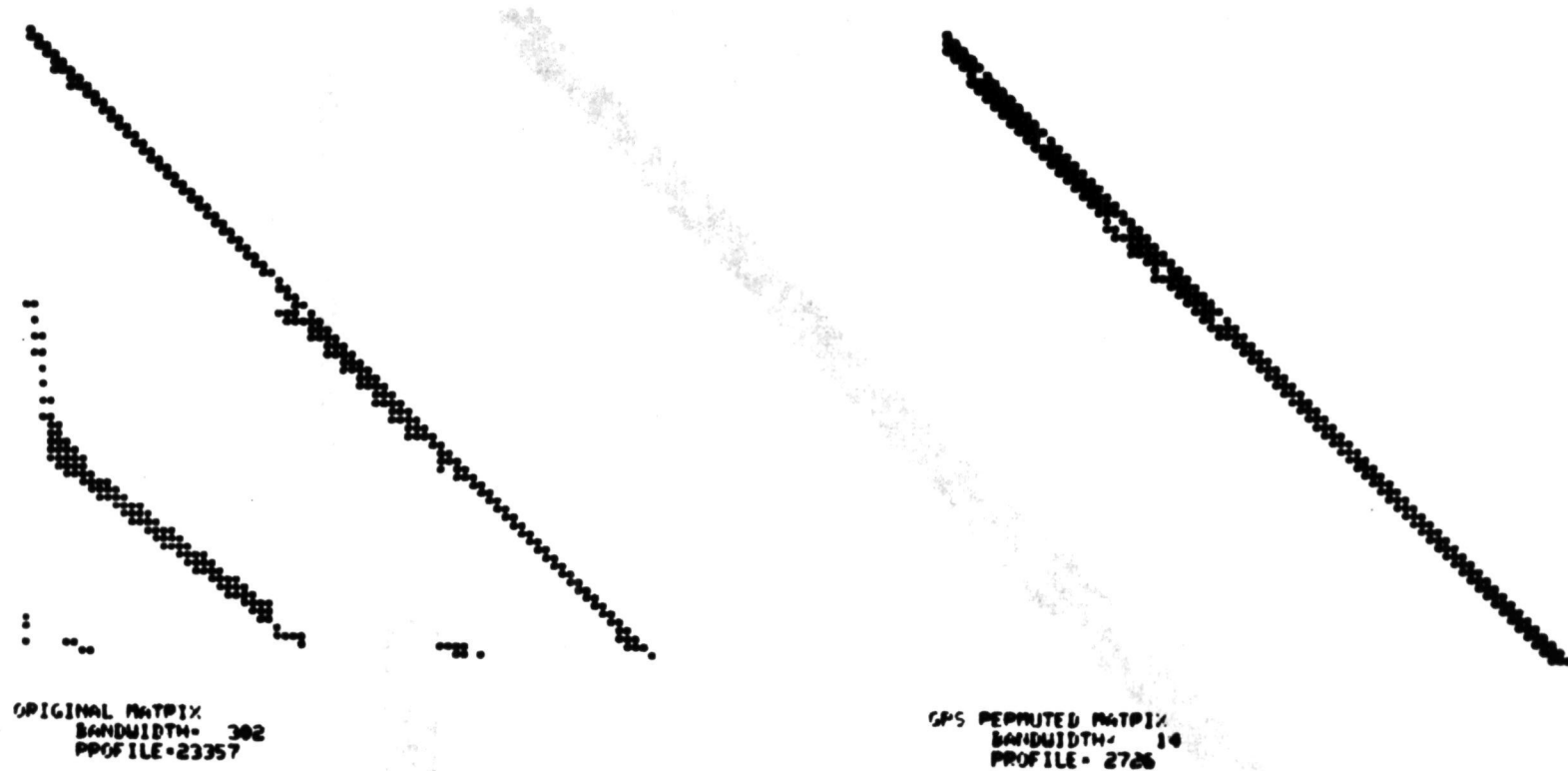


Figure 4.- Display of lower triangular halves of matrix before and after renumbering.

## SPACEBAR

### KINEMATIC DESIGN BY COMPUTER GRAPHICS

Richard J. Ricci

Automation Development and Loft Group  
Lockheed-California Company  
Burbank, California

32

#### SUMMARY

The development of kinematic mechanisms on the design board is often an extremely complex and time-consuming task. It was this impetus which led to the development of the interactive graphics program SPACEBAR at Lockheed.

The SPACEBAR program allows the direct design and analysis of mechanisms right at the terminal screen. All variables including linkage geometry, stiffness, and applied loading conditions can be input and/or changed from the terminal. All data can be described and displayed in three dimensions. All mechanism configurations can be cycled through their range of travel and viewed in their various geometric positions, again in three dimensions. Output data includes geometric positioning in orthogonal coordinates of each node point in the mechanism, velocity and acceleration of the mechanism node points throughout its range of travel, and internal loads and displacements of the individual node points and linkages. All analysis calculations are performed at the scope and take, at most, a few seconds to complete. Output data can be viewed at the scope and also printed at the discretion of the user.

#### INTRODUCTION

The designing of light weight kinematic mechanisms such as control systems, landing gears, and door hinges is a major area of development in the design of aircraft and their subsystems.

Developing these complex mechanisms and control systems and predicting their motion is and always has been one of the more challenging tasks for the design engineer. Once beyond two-dimensional kinematic motion, geometric design techniques quickly increase in complexity and the chances for system inaccuracies multiply. The board work required for even relatively simple designs becomes a maze of overlays upon overlays, and development time can turn into weeks and even months. Often, complicated analytical calculations to determine loads and particularly deflections, are nearly nonexistent until the final design stage has been attained. As a result, it is usually too late in a schedule to make the changes desired to optimize the weight-load relationship.



A new method was needed to overcome this labyrinth of drawings and calculations and to speed up the iterative procedure for arriving at a final design. A way in which periodic checks of loads and deflections could be made without consuming days of development time was the basis that led to the development of SPACEBAR in computer graphics at Lockheed.

### Computer Graphics Development

Lockheed began investigating the applications of computer graphics to design and manufacturing in the 1960's. It was soon apparent that computer graphics could reduce development time and cost, and alleviate many of the engineer's tedious everyday tasks. In the business world, this had been the main thrust of computer development, but in engineering, the emphasis had been concentrated on solving the bigger and more complex analytical problems. The everyday task was still a millstone around the engineer's neck. With the advent of interactive CRT's and the proper software, however, Lockheed felt that many of these chores could now be automated.

### The Development of SPACEBAR

Mechanism development was one of these chores; a design task which was fairly simple conceptually, but which consumed a great deal of time and effort in developing a solution. The program developed to help alleviate this problem was called SPACEBAR.

SPACEBAR is a three-dimensional kinematics design and analysis program developed specifically for computer graphics at the Lockheed-California Company. It was created to help accelerate and improve the design of mechanical systems, thereby reducing development cost and achieving a more optimal product. Although it was developed with an emphasis on aircraft systems, its capabilities allow the design of a large variety of mechanisms including such items as control systems, landing gears, door hinge mechanisms, and packaging equipment. In fact, nearly any mechanical system which includes the use of linkages, disk cams and tracks can be designed, at least in part, on SPACEBAR.

SPACEBAR is an interactive computer graphics engineering program. It enables the user to vary any system parameter at will, and to evaluate the newly developed configuration, as displayed on the terminal screen, in a matter of seconds. The program allows the modification of all mechanism geometric parameters such as link lengths, fixed axle locations, rider point locations, and cam and track curve descriptions. This manipulated data is then used to generate a new geometric model. The program also allows the modification of all analytical stiffness and strength data associated with the physical mechanism. From this design data the program calculates and displays all the geometric and analytical information necessary for interactively designing kinematic systems. It determines system motion, internal member loads and system deflections, and displays this data in the physical relationship of the mechanism in three dimensions.

## HARDWARE

To operate the SPACEBAR program Lockheed uses either an IBM 2250 or a Vector General graphics terminal on-line with an IBM 360-91 mainframe computer. A Data General Nova 800 is used to provide compatibility between the Vector General terminal and the IBM 360-91 mainframe. The mainframe computer is used for multiple tasks and the program operates in a 130k dedicated core space with a very high priority.

The graphics terminal provides three modes of communication between the user and the program (see Figure 1). The first of these is through the Function Key Box. The program can test which of 32 independent buttons contained in the function key box has been pressed and from this determine a course of action. This method is often used to bring up new subroutines contained within a program. In SPACEBAR the function key box is used solely for rotating the three-dimensional stick representation of the model being investigated. By pressing one of six active function key buttons, the user indicates to the program the desired direction of model rotation. Two buttons are used for rotation (+ or -) about each of the system orthogonal axes. The second mode of communication with the program is the typewriter keyboard. Through this medium the user can enter any alphanumerical information desired. SPACEBAR uses the typewriter keyboard for entering mechanism geometric data, applied loads, and stiffnesses. The third method of communication is with a fiberoptic light-sensitive pen. In SPACEBAR the light pen is used to select data points to be changed or one of a "menu" of options to be activated. The options contain the many different analytical and geometrical subroutines which the program uses to evaluate the mechanism model.

Interactive response while operating SPACEBAR is very rapid; usually less than one or two tenths of a second. The only exception to this is during the generation of the inverse matrix used to determine system loads and deflections. This calculation can sometimes take up to several seconds for completion depending upon the mechanism being investigated.

The SPACEBAR program itself is one of several analysis programs contained in specified load modules. These load modules are stored on disk packs which can be routinely accessed from the terminal.

## HARDCOPY OUTPUT

Both microfilm plots and standard printout can be generated from the SPACEBAR program. In the analysis phase of the program, printout is generated automatically. In the geometric phase of the program, using the light pen to detect a screen menu labeled "PRINT" will generate the desired printout. For microfilm plots the user must first inform the computer operator that plots are to be initiated. The operator then loads a plot tape on a tape drive and this tape drive is dedicated to the SPACEBAR program during its operation. The user is then free to select that information for which plots are desired. By using the light pen to detect a screen menu labeled "PLOT", microfilm plot data, duplicating the picture shown on the terminal screen, is output onto

the dedicated plot tape. Upon completion of the session's work, the data on the plot tape is converted into microfilm from which hardcopy can be generated using a Xerox process.

### CAPABILITIES

The analysis capabilities of the SPACEBAR program are extensive and provision is made to query virtually any parameter of the mechanism the user may wish to study. The types of parameters to be studied can be broken down into two basic sets: geometric or motion parameters, and analytic or structural parameters.

#### CAPABILITIES - GEOMETRIC

The SPACEBAR program has the geometric capability of integrating into a unified three-dimensional mechanical system such diverse mechanical devices as the 4-bar linkage, either by itself or with a rider point, the cam with cam follower, and the track. It can then display the operational characteristics of this integrated system on the terminal screen. Finally, it can store this mechanical system for later retrieval and study.

#### Model Description and Restrictions

The basic units for any mechanism used in the SPACEBAR program are the Fourbar linkages (Figure 2). All other mechanism devices are "attached" or interlinked with these. In any single case the program can handle up to four interlinked fourbar linkages. Rider points can be attached to any or all of these.

A given fourbar linkage may be "tied" to the rider point of the previous fourbar by use of a "tie" switch. This will tie the motion of the following fourbar to the previous linkage as follows: the rider point of the previous linkage is analogous to end point of the first bar of the following linkage (see Figure 3). In effect this results in a simulated variable length first bar in the second fourbar linkage.

Disk cams and cam followers are interlinked to fourbar linkages by use of the "cam insert" feature (Figure 4). The only constraint on cams is that they cannot be free to move in space but must rotate about a fixed axle point. All cams and cam followers are described by a series of points to which a spline curve is then fit. The spline fit curve is based on a piecewise cubic with continuous first and second derivatives. Each mechanism in SPACEBAR can have up to four cams. In the analysis portion of the program each cam--cam-follower arrangement is analyzed as an instantaneous fourbar at a given angular position.

Tracks in SPACEBAR are defined by the motion of a link end point on a fixed track centerline. The track centerline can be either straight or curved in three-dimensional space and is represented by the intersection of two surfaces (e.g., the intersection of two planes results in a straight line).

Fourbar linkages are "tied" to tracks in the same manner as they are tied to rider points, except in this instance it is the end of the second link in the preceding fourbar which is tied to the track (see Figure 5).

#### Data Input

Data required to establish a mechanical system in SPACEBAR can be entered in two ways. The first is by submitting a punched card deck to a batch program. The batch program then enters the mechanism's parameters onto a data set accessible by the program. For any new system with an extensive and complicated linkage, the submitting of axle location, link lengths, and stiffness data by punched cards will save a considerable amount of terminal usage time. A manual has been developed for the user which describes the method and order of data submittal.

The second method of entering data is through the terminal itself. Selection of the menu "INPUT" (Figure 6) will bring up the input data display for the mechanism under investigation (Figure 7). Any data point in this menu can be changed by simply light pen selecting that value and keying in a new value on the typewriter keyboard. This method of entering data is particularly useful for refinement of the mechanism and the resulting case can be used instantly at the terminal or saved on a disk pack for future reference.

#### Display Modes

Two display modes are available in SPACEBAR to help the user attain a better understanding of the mechanism that is being created. The first is a display of the mechanical system showing both the plan and side views (Figure 6). Included in this display are the X, Y, and Z coordinates of each point in the model, the angle through which each axle in the mechanism rotates with respect to its reference position, the initial and final input and output angles through which the mechanism is desired to traverse, and the delta angle through which the mechanism will step when cycled. Also included in this first display is a graph depicting the desired (and after the mechanism has been cycled, the actual) input-output angular motion relationship.

The second display mode depicts the mechanism in a three-dimensional axenometric projection (Figure 8). This three-dimensional stick model representation may be rotated about any of three orthogonal axes until the desired projection is achieved. The rotation can be accomplished either by using the function key box or by keying in the desired angles on the typewriter keyboard. A magnification option exists which allows the user to blow up any desired portion of the mechanism for a more detailed examination of its operation. In both of these displays the mechanism can be cycled or stepped through its desired range of motion with the pictorial presentation of the mechanism changing accordingly. If the mechanism is unable to traverse the given range of motion, the system will stop at the last attainable position and a note will appear indicating an impossible geometric relationship has been asked for.

## Geometric Data Output

The methods developed to enter and change data in the SPACEBAR program have proven extremely useful to the design and analysis engineers. It allows them the freedom to check and modify the geometric characteristics of any parameter of the system at will and in a few minutes of time.

A user can cycle the particular system, check out his geometric and analytic characteristics in seconds and determine any deficiencies. Then, if deficiencies or improvements are determined, corrections can be made right at the terminal, again in seconds. The new system model is immediately available to be analyzed. The whole process from system modification to geometric and analytical checkout takes but a few minutes of terminal time.

The type of geometric data output on SPACEBAR includes all system nodal point locations, nodal point velocities, and nodal point accelerations. This data is available to the user at any geometric positioning of the mechanism within its range of travel. There are two ways in which this output data is displayed. The first is located within the plan and side view mechanism geometry display discussed above under display modes (Figure 6). It includes the X, Y, and Z coordinates of each nodal point in the system. These values change accordingly as the mechanism is cycled or stepped through its range of travel and always records the current geometric position of each nodal point during the cycling sequence. Along with the X, Y, Z coordinate display is a graph depicting the relative motion of the mechanism input angle vs. its output angle. An enlarged view of this graph is available under a menu option labeled "GRAPH" which the user may select. The input-output graph can contain up to three separate curves depending on the user's needs. The first curve is empirical data that is input by the user and represents the desired relationship between the input and output angular motion of the system under development. This data can be input by punched cards or at the terminal. The second curve is calculated internally by the program as the mechanism is cycled and represents the actual relationship of the input vs. output angle. The third curve is generated when the mechanism is cycled again. This is usually done after the mechanism has been modified and thereby allows the user to compare the motion of the last two developed mechanisms with the desired input curve. Each time the system is cycled the oldest of the two internally generated curves is erased and a new curve based on the motion of the currently displayed mechanism is generated.

The second method of displaying geometric output data is contained in a display labeled "VARIABLE SELECTOR" which is similar to the geometry display. This display is used to compare the relationship between various geometric parameters while the mechanism is cycled through its range of travel. The relationship is presented in the form of a graph at the top half of the terminal screen. The variables to be compared are selected by the user through the use of the light pen and include the X, Y, or Z of the angular positioning, velocity, or acceleration of any nodal point in the mechanism. The user may change these parameters at will and obtain a new graph.



## Data Storage

Data entered either by punched cards or at the terminal can be stored in the SPACEBAR program. Two separate files exist for this purpose. Only card input is stored on the first file. The user accesses this file by light pen selecting the option "NEW CASE" available on most of the SPACEBAR displays. Changes can be made to a case from this file and the modified case may be saved in the second file. The second file is accessed by light pen selecting the word "FILE". The user may delete, modify, or add cases to this second file. There also is a hidden code name which may be added to cases in the second file to prevent unwanted modifications. Each of these two files can store up to 100 different cases (200 cases total).

## CAPABILITIES - ANALYTIC

Once the geometric parameters of a mechanism have been established, the user proceeds to the analysis portion of the program where an analytical or structural evaluation of the system can be made. The analysis portion is divided into two parts. First is the determination of linkage internal loads and system support point reactions from externally applied loads; second is the calculation of system deflections due to externally applied loads. Two types of external load conditions can be applied to a given mechanism in SPACEBAR: jam loads and applied loads.

### Internal Loads - Jam Loads

The jam load condition consists of fixing the mechanism from rotating at the last axle point and applying a 112.98 m-N (1000 in-lb) moment at the first axle point. Since there is only one load path possible the mechanism is always determinant. The internal loads are then calculated by simple structural joint vector analysis. The entire process is accomplished by a geometric matrix which is automatically set up by the SPACEBAR program and which varies according to the size and component makeup of the mechanism. This geometric matrix is then multiplied by a special jam loads matrix to determine the system internal loads.

The user has the option of asking for two sets of jammed loads answers. The first set of answers consists of the load in each link and at each support point, the moment at each intermediate axle, and the input-output mechanical advantage of the mechanism at the selected angular position. This data is displayed in tabular form under the "Internal Loads - Jammed" menu of the program (Figure 9). The second set of answers consists of a plot of the load in a given link or at a selected support point vs. the mechanism input angle over the range of travel requested. This graph is displayed under the menu "JAM LOAD SELECTOR" which is similar to the geometry display and allows the user to select the particular parameter to be investigated.

### Internal Loads - Applied Loads

The applied loads conditions consist of fixing the mechanism at the first axle point and applying loads wherever desired throughout the mechanism. Provision is made under the applied loads menu (Figure 10) to add or delete

a load at any system node point by light pen selecting the node desired and keying in the load. Loads are applied along the basic X, Y, Z orthogonal axis of the mechanism. The internal loads for the applied loads condition are calculated similarly to that of the jam loads condition. A geometric matrix is constructed which resembles that of the jam loads geometric matrix and an applied loads matrix is multiplied by that geometric matrix to determine the internal loads. Again, this is all done internally by the program. Resultant internal loads are displayed on the "Internal Loads-Applied" menu of the program and resemble very closely that of the "Internal Loads-Jammed" menu.

### System Deflections

The deflections of a selected mechanism are generated similarly to that for the internal loads condition. In this circumstance, however, the system stiffness parameters as well as the geometric parameters and applied loads must be established. Linkage areas and inertias as well as support point spring rates must be supplied for the given mechanism. This data is entered either by punched cards or at the terminal under the two displays "AREA AND INERTIA" and "SPRING SUPPORTS" (Figures 11 and 12). Data submitted by punched cards must be entered through the batch program, while data submitted at the terminal uses the light pen and typewriter keyboard. Once all required stiffness data has been entered into the program, the deflection calculations can be made. The user has the option of calculating two different sets of deflections. The first deflection set is based on a mechanism jammed at the last axle with a 112.98 m-N (1000 in-lb) torque applied on the first axle. This will give the user the total stiffness of the mechanism being investigated as well as display the X, Y, and Z deflections of each individual node point (Figure 13). The second set of deflections will be based on any applied loads which were entered under the applied loads menu of the program. It also displays the deflection of each individual node point.

In all the calculations discussed above, the user may generate the prescribed data in any mechanism position he chooses. He merely sets the linkage position in the geometry section of the program and then asks for the data in the analytic section of the program.

### Internal Program Operation

To calculate the analytical data described above, SPACEBAR sets up internally and automatically a series of finite element type flexibility and stiffness matrices. A stick model detailing the system and element members of a typical mechanism is shown in Figure 14. Once the basic flexibility matrix has been established, it is multiplied by the applied loads or jam loads matrix to determine the resultant system loads and deflections.

$$[f] = [E]^{-1} [AL]$$

$$[u] = [K]^{-1} [AL]$$

where: E = geometry matrix  
 K = stiffness matrix  
 AL = applied loads matrix  
 u = nodal displacements  
 f = nodal internal loads

The largest matrix which an individual SPACEBAR case can generate is of the order of 60 x 60. This is equivalent to four interconnected fourbar linkages, each with an independent rider point. SPACEBAR also, however, has the capability of transferring output data from one case as input data into another case. In this way very large mechanisms can be handled piece by piece.

#### Present Usage

SPACEBAR's primary usage to date has been to check and verify the operating characteristics of several existing mechanical systems. These include simulation of the Navy S-3A horizontal stabilizer tab actuation mechanism, simulation of an airstairs for the L-1011 wide body TriStar airliner, and modification to the geometry of the S-3A landing gear door mechanism. In the last case, it was estimated that at least 10 hours of board design work was done in one and one-half hours of terminal time with the internal loads and deflection analyses as a bonus derived from using SPACEBAR. Correlation of the data from SPACEBAR with previous manual methods was found to be very accurate and served to verify the program.

#### FUTURE ENHANCEMENTS

The SPACEBAR program, while a complete and useful tool in its present form, is still in its mechanical infancy. The addition during this last year of such features as cams, tracks, and a rider point to the basic fourbar linkage has greatly increased its versatility, and the inclusion of velocity and acceleration calculations along with some new data output displays has added to the system's data retrieval capabilities. However, there remains a multitude of major types of mechanical motions which are still beyond the program's existing analysis techniques.

As SPACEBAR gains in usage and importance, more of these complex features will be developed and incorporated. Mechanical devices simulating idlers, floating links, and variable length links controlled by spring stiffeners are all possible future enhancements. Actuators along with actuation motions and stiffnesses are also contemplated.

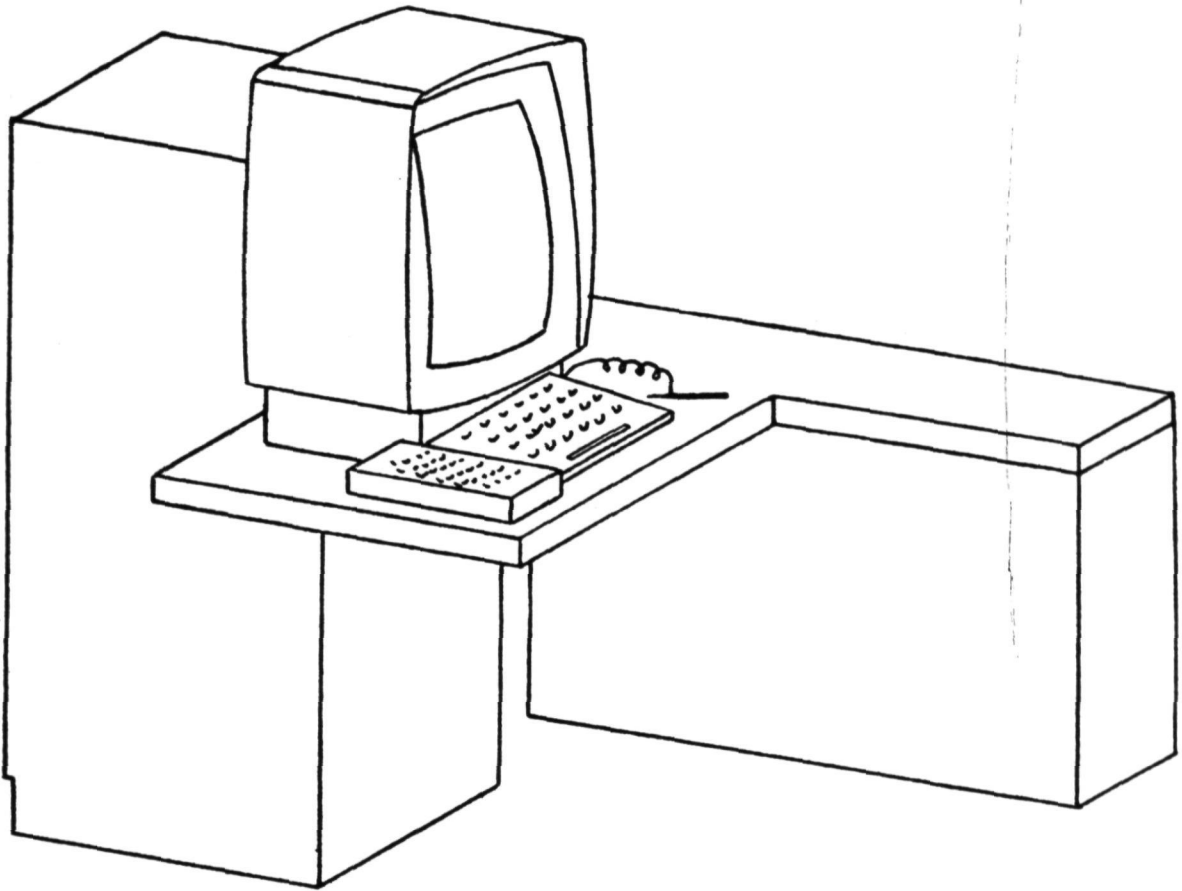
Other enhancements which look promising are a direct interface with CADAM (Computer Graphics Augmented Design and Manufacture), Lockheed's graphics drafting package, allowing the fast and efficient transfer of mechanism geometrical parameters from one system to another and finally a direct hookup to a computer batch program where vibration and stability investigations can be carried out.



All of these enhancements will increase the versatility and capability of SPACEBAR with the eventual goal of designing and analyzing most, if not all, of the basic kinematic mechanisms by interactive computer.

#### ACKNOWLEDGEMENTS

I wish to acknowledge the contributions of Rick Fromm and Sy Southworth for the development of the original fourbar mechanism in SPACEBAR, and the contribution of Bridget Shycoff for her programming effort in the subsequent additions of the rider point, cam, and track mechanisms, and in the development of the velocity and acceleration subroutines.



From left to right in the work area in front of the  
Cathode Ray Screen:

1. Function Key Console
2. Typewriter Console
3. Light Pen

Figure 1 - The IBM 2250 display unit.

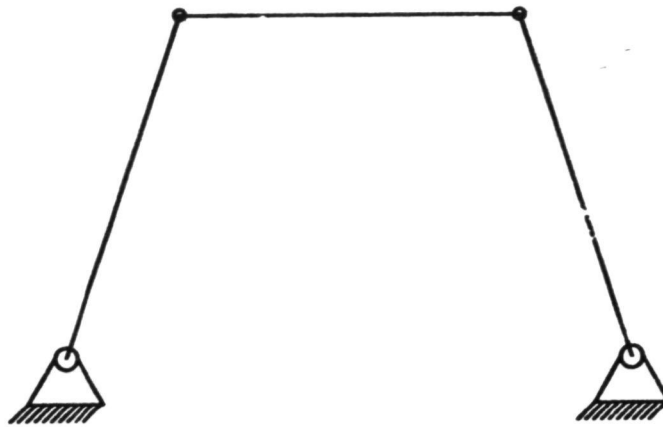


Figure 2 - Typical fourbar linkage.

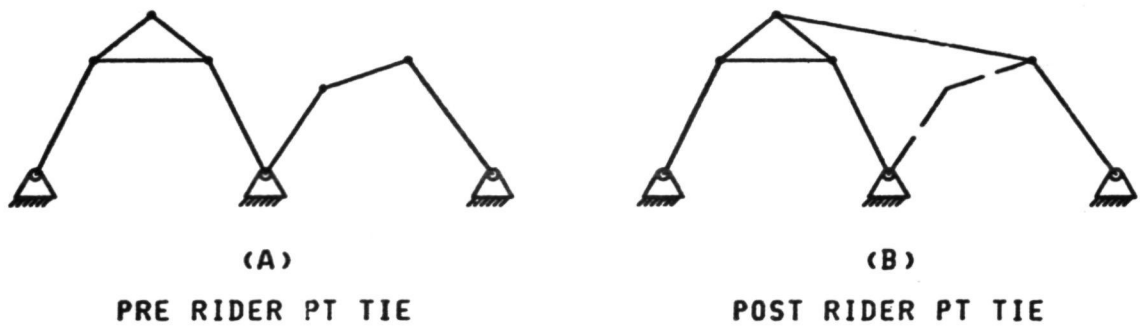
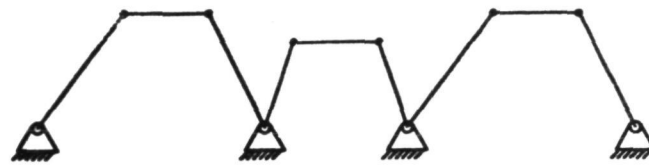
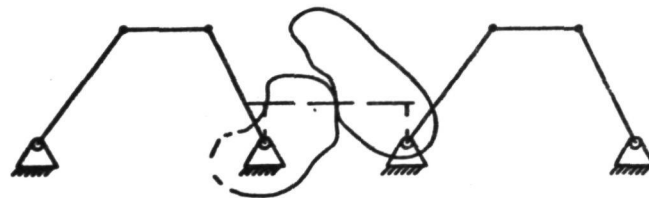


Figure 3 - "Tie" mechanism attaching following fourbar linkage to previous fourbar linkage rider point.



(A)

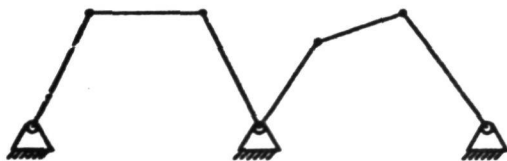
PRE CAM INSERT



(B)

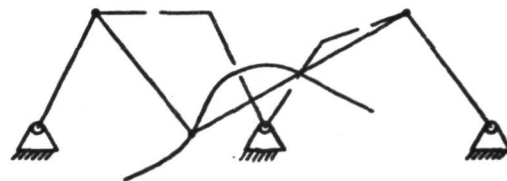
POST CAM INSERT

Figure 4 - "Tie" mechanism converting fourbar linkage into cam actuation device.



(A)

PRE TRACK TIE



(B)

POST TRACK TIE

Figure 5 - Track "tie" mechanism attaching fourbar linkages to track centerline space curve.

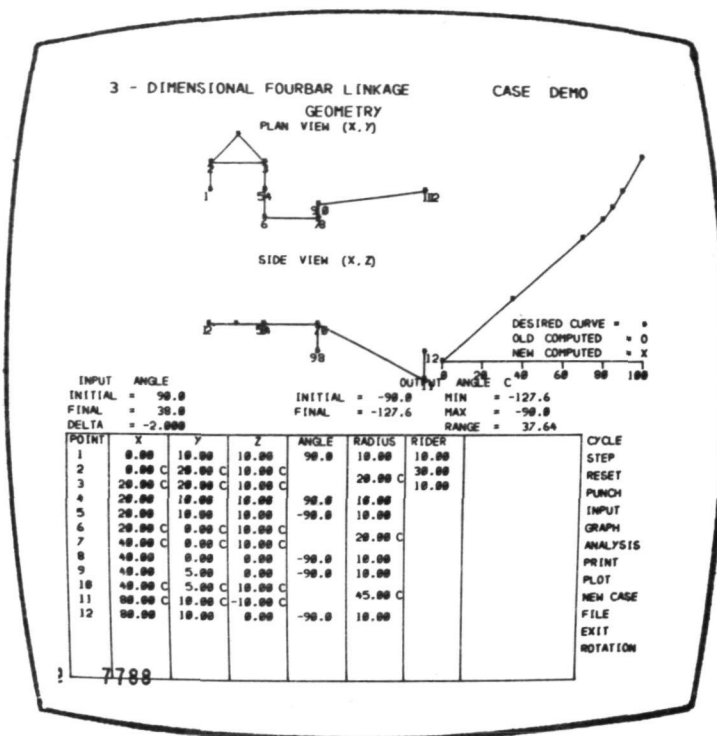


Figure 6 - Geometry display - note list of menus along right hand side of screen. Light pen selection of these menus accesses various routines in the program.

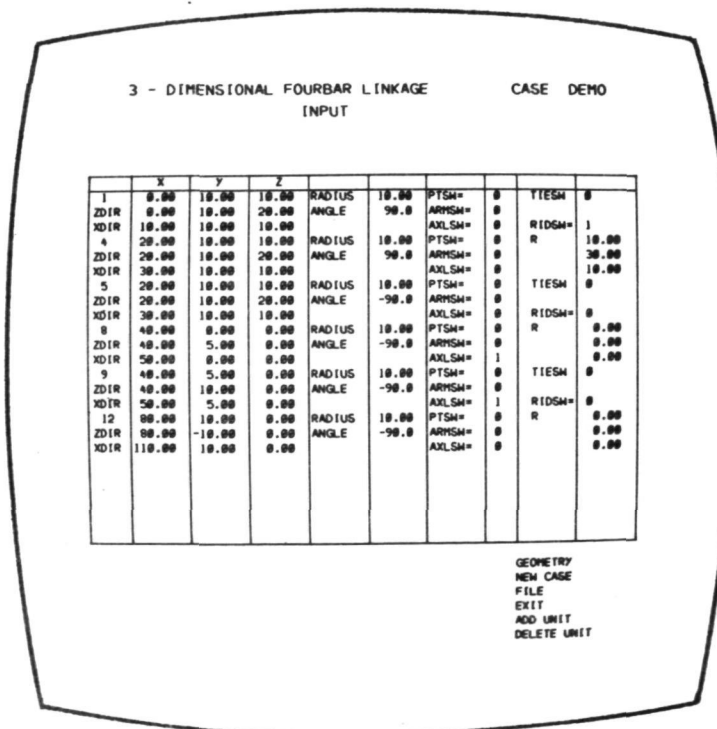


Figure 7 - Input display showing typical data description required for a given mechanism.

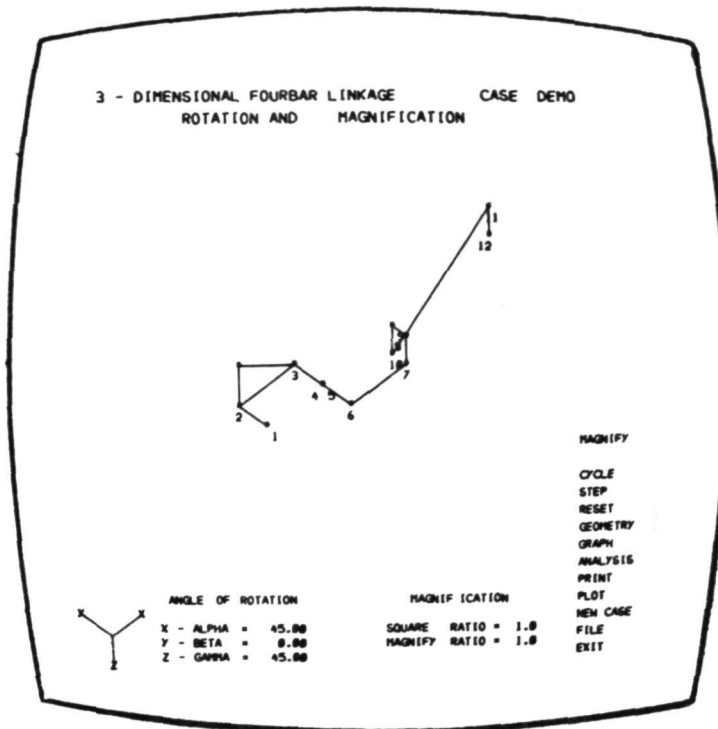


Figure 8 - Rotation display depicting axenometric view of given mechanism.

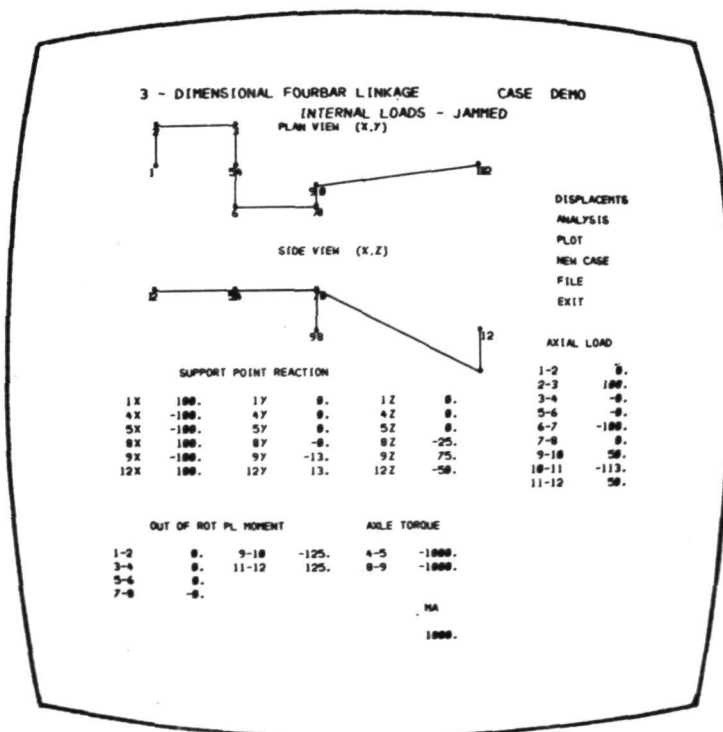


Figure 9 - Internal loads - Jammed menu showing mechanism internal loads.

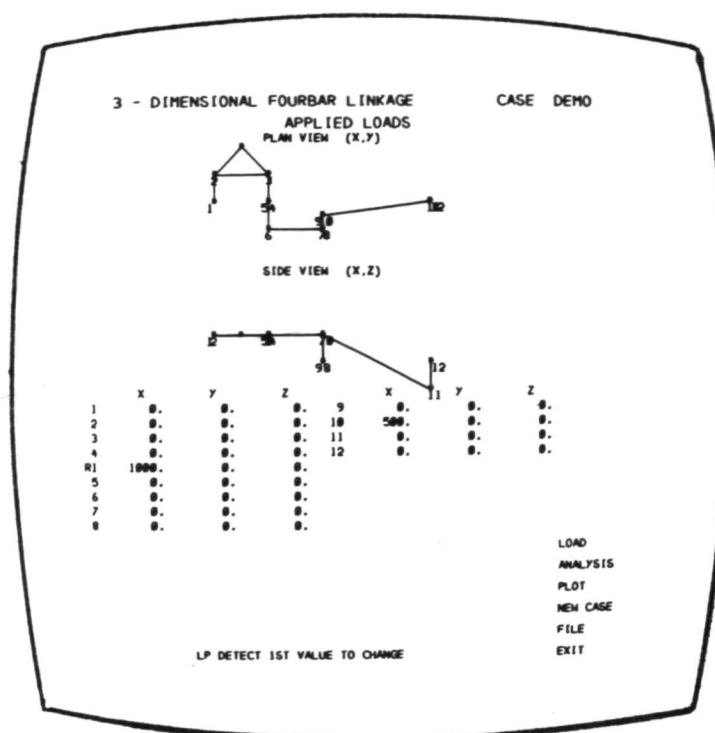


Figure 10 - Applied Loads menu showing mechanism and applied loads.

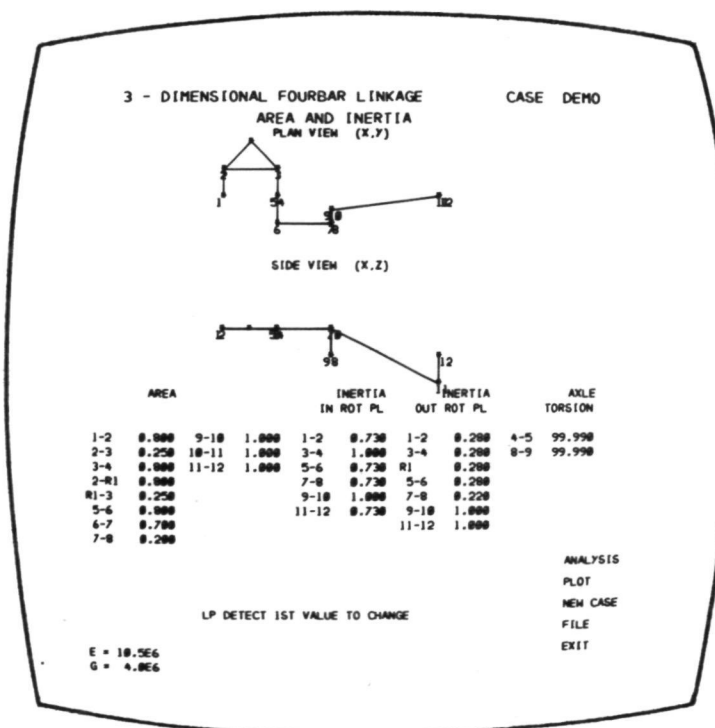


Figure 11 - Area and Inertia Menu - displays the stiffness parameters of the given mechanism.

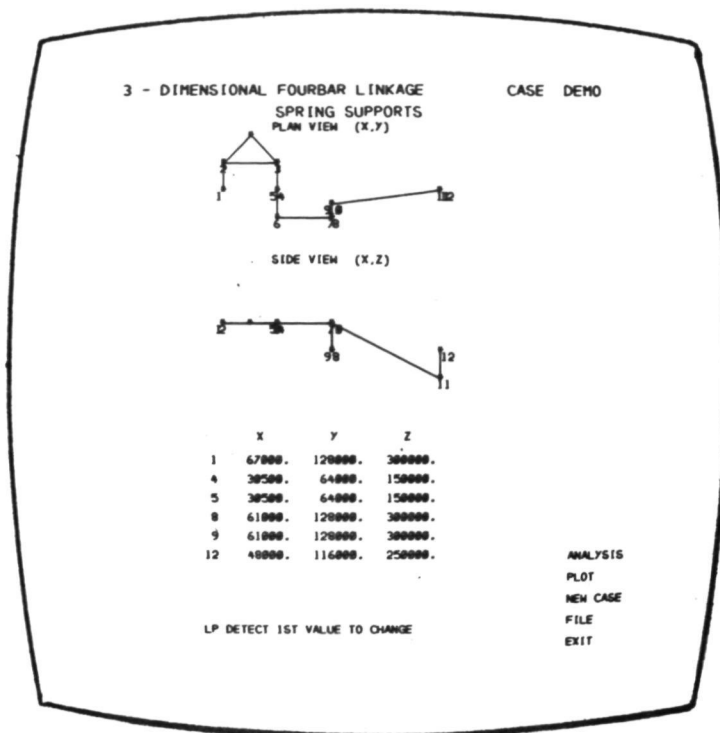


Figure 12 - Spring Support Menu - shows the X, Y, and Z spring supports on the fixed position nodes of the mechanism.

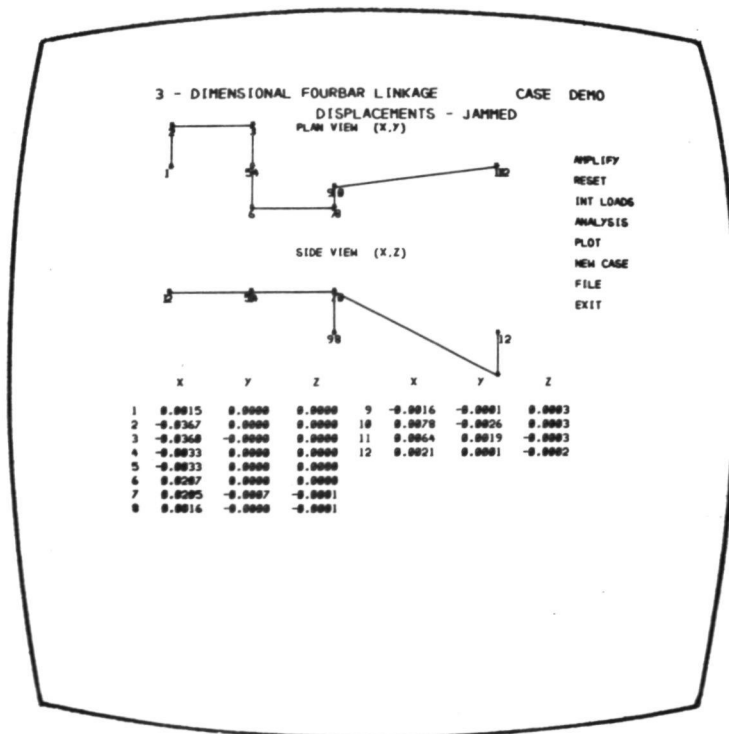


Figure 13 - Displacement Display showing internal displacements of each node point in model.



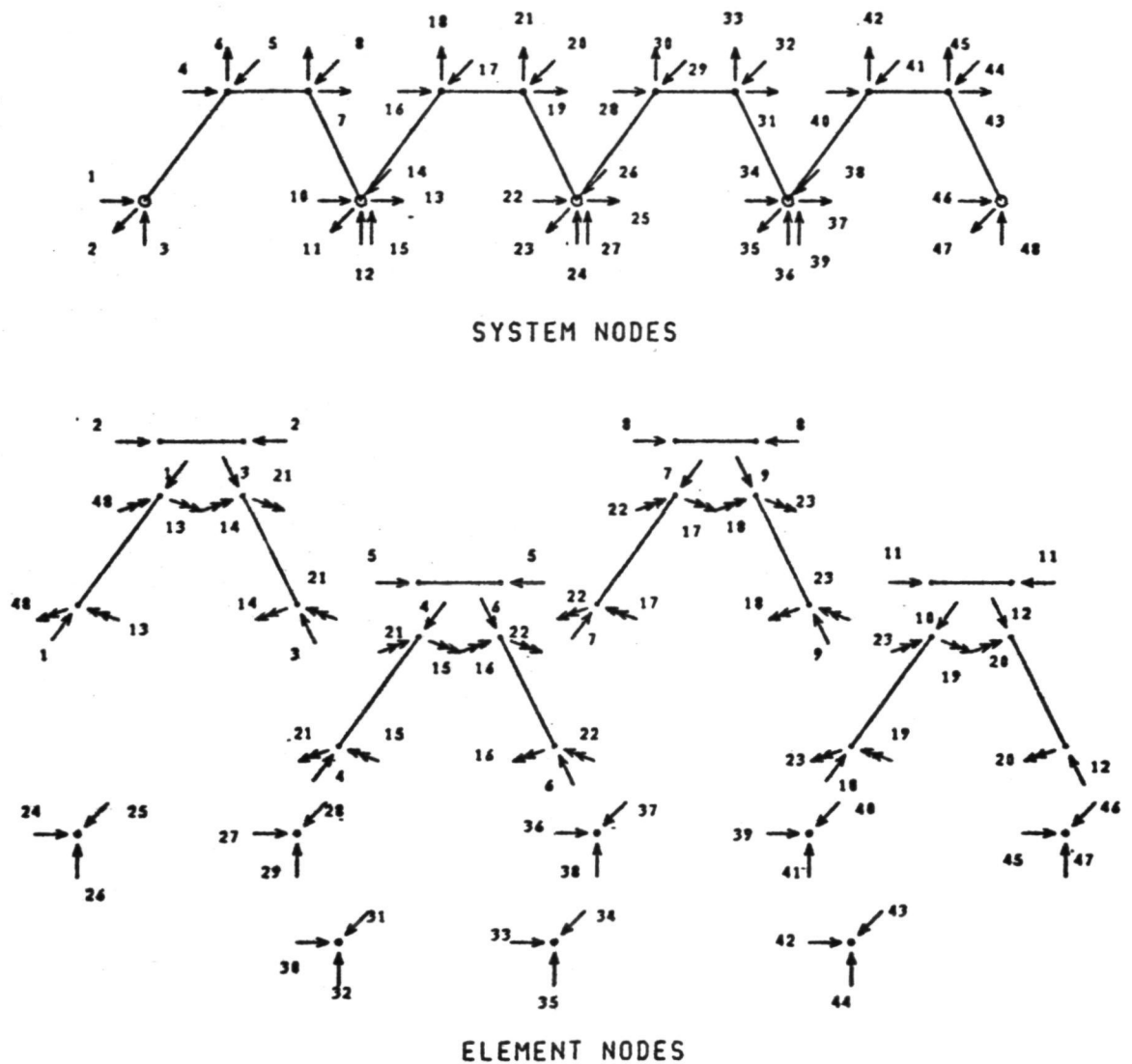


Figure 14 - Stick model detailing finite element system and element node relationship.

## SIKORSKY INTERACTIVE GRAPHICS SURFACE DESIGN/ MANUFACTURING SYSTEM

33

Richard Robbins  
Sikorsky Aircraft

### INTRODUCTION

The design, analysis, and manufacturing of aircraft components using free form surfaces is a complex task. In the late 1960's and early 1970's Sikorsky Aircraft Engineering Design and Manufacturing Engineering Branches were using APT FMILL in an attempt to accomplish this task. Limitations of APT FMILL, problems with transfer of data between departments, and the obvious problem of surface evaluation in a batch mode led to the development of Sikorsky's Interactive Graphics Surface Design/Manufacturing System.

In addition to the basic surface definition and viewing capabilities one would expect to see in a system such as this, numerous other features are present: surface editing, automated smoothing of control curves, variable milling patch boundary definitions, surface intersection definition and viewing, automatic creation of true offset surfaces, digitizer and drafting machine interface, and cutter path optimization. Documented costs and time savings of better than six to one are being realized using this system.

- This system was written in FORTRAN and GSP for use on IBM 2250 CRT's; Sikorsky currently has three 2250 mod III CRT's running on an IBM 370/158.

## SURFACE

The surface is defined from a rectangular array of points, each column and row of which describes a cubic control curve. The elementary surfaces, bounded by four intersecting control curves, are mathematically defined by the four corner points and the two tangents at each of those points.

In general, the control curves are 3-D space curves. However, in our experience it has been found for ease of surface development that the control curves in one of the parametric directions be in plane.

## SURFACE DESIGN DEVELOPMENT

### Point Generation

The primary input to the surface system is a set of control points that define the surface control curves. The number of control points required and the spacing of these control points are the variables that the surface designer must work with to obtain a desired surface.

For example, if the curve spacing is uneven, the resultant surface may have glitches, making the surface unacceptable when milled. It is therefore important from the designer's point of view that the control curves can easily be generated or modified, and the surface checked.

The Surface Design System provides several algorithmic tools in the designers behalf for generating control points. The methods used to generate the 3-D control points are:

- (1) A skeleton set of 2-D points obtained from digitizing known contours or by APT programming is read into the Surface Design System. The points are curve fitted using a composite of straight lines, conics, and splines. These planar lines now become the basic reference curves for the surface. 2-D points are generated from these curves (by intersecting with planar curves or by using the equally spaced point generating algorithm for a curve) and then transformed to 3-D space to obtain the control points.

- (2) Projecting control curves onto coordinate planes and interpolating the projected curves at a specified value to obtain new control points.
- (3) Intersecting the control curves of a surface with a plane.

All points may be observed visually in the Surface System and may be modified as required. For instance,

- (1) Points may be deleted
- (2) Coordinates of points read out
- (3) New points keyed in
- (4) Multiple points may be defined at a coordinate location
- (5) Surface points may be replaced
- (6) Points may be transformed

### Tangent Generation

At the heart of the surface generation section of the program are algorithms to define and edit tangent vectors at control points. The algorithms may be applied to all points on individual curves, or at all points on all curves, within a section of the surface, in either or both parametric directions.

Tangent generation algorithms may be selected from -

- (1) Algorithms based on segment chordal distance.
- (2) Algorithms based on 2nd derivative continuity between segments of control curve.
- (3) Algorithms based on matching a 2-D spline through a set of control points. The control curve is projected onto a working plane. This projected curve is then matched, if possible, against a 2-D spline fitted through the set of projected control points. For example, the tangent vector direction of the control curve is matched against the tangent vector direction of the spline. The magnitude is matched, where possible; otherwise, it is averaged.

## Surface Checking

A major problem with developing a surface is in checking the validity of the surface. Procedures to help solve this problem are

- (1) New planar control curves may be checked automatically by projecting them onto a working plane and comparing with 2-D splines through the sets of projected control points
  - (a) Checking may be done visually by comparing the overlaid curves
  - (b) Automatic checking may be done by an algorithm which indicates those segments that are out of tolerance by more than a specified amount. Tangents at the segment end points can then be edited individually until segments are within tolerance
- (2) Displaying and plotting plane intersections of the surface
- (3) Magnifying control curves in one direction in order to detect curve irregularities
- (4) Surface continuity may be checked by displaying a parametric distribution of connected points in either of the surface directions. Since the parametric distribution simulates the cutter path when the surface is milled, it can be used to identify possible problem areas in surface definition.

## MILLING DESIGN

One of the problems in the past with milling the surface - a batch postprocessing operation - was limiting the milling boundaries to the input geometric control curves. The problem for the user was introducing control curves into his model often with adverse results, solely for the purpose of defining milling boundaries.

The graphics program allows user to define milling boundaries after the surface has been defined. The boundaries can be defined from

- (1) Any parametric curves including control curves
- (2) Plane/surface intersections
- (3) 2-D projections of straight lines, conics, splines onto surface

The post-graphics processing performs

- (1) Chord and scallop height analysis on the milling patch to determine the optimum number of cutter paths and point to point moves required for a specified tolerance
- (2) Cutter location data generation
- (3) Machine dynamics and paper tape generation.

## APPLICATIONS

### Helicopter Blades

The primary use of this system has been in the design of helicopter blades and manufacture of blade tooling. Contours which have been defined from aerodynamic considerations are used as a basis for defining surface control curves. Intersecting the developed blade surface with planes and comparing the intersection contours with known cross sections is used as a measure of blade acceptability. Key features of the Surface System in blade design and milling are

- (1) True surface offset capability which is frequently required in blade development
- (2) Independent construction of milling patches for blades. This frees the blade designer from milling considerations during surface modeling of blade.

## NASTRAN Grid Point Generation

The surface System has been found useful in generating grid points for finite-element programs such as NASTRAN. Most of the nodal points required in modeling a helicopter fuselage, for instance, are found in the body or tail section - surfaces of constant cross section, ruled surfaces, or transition surfaces between ruled surfaces. These are all points which can easily be defined. For example, contour points describing intermediate frames which may be canted are computed by the plane/surface intersection algorithm. The transition surfaces are defined by generating the body and tail sections separately and allowing the resultant boundary conditions to define the transition surface. (See figs. 1 to 3.)

## CONCLUDING REMARKS

Although there are disadvantages mathematically with this surface under certain conditions - zero twist vectors and triangular surface segments for example, and there are limits to the surface checking that can be performed interactively, the system has resulted in considerable cost savings for many applications.

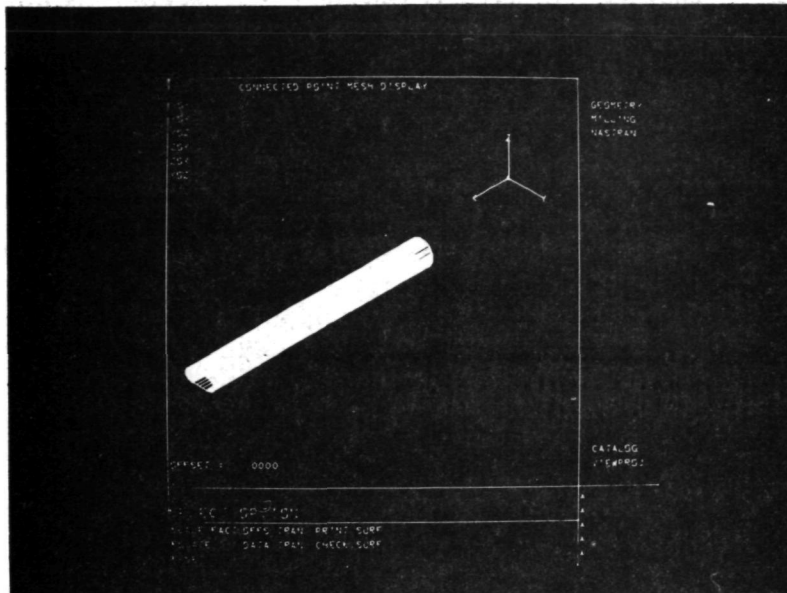


Figure 1.- Orthogonal view of blade spar.

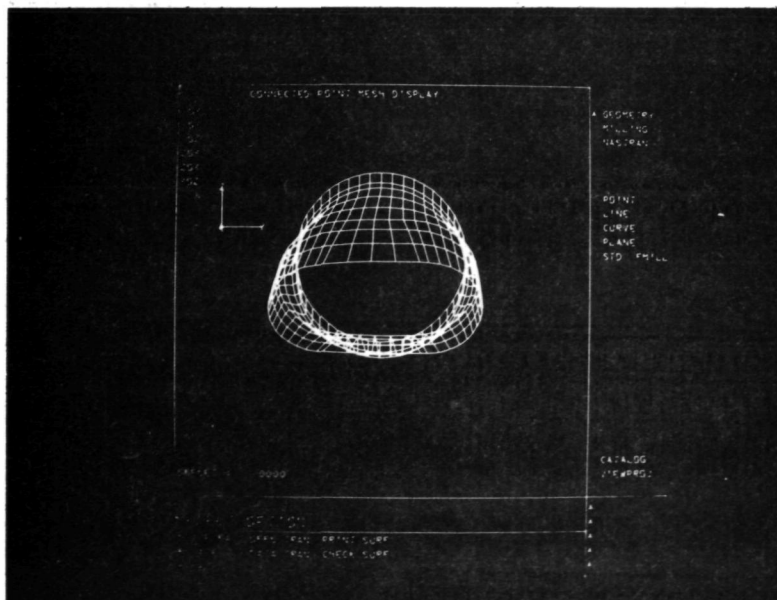


Figure 2.- Blade spar viewed end on.



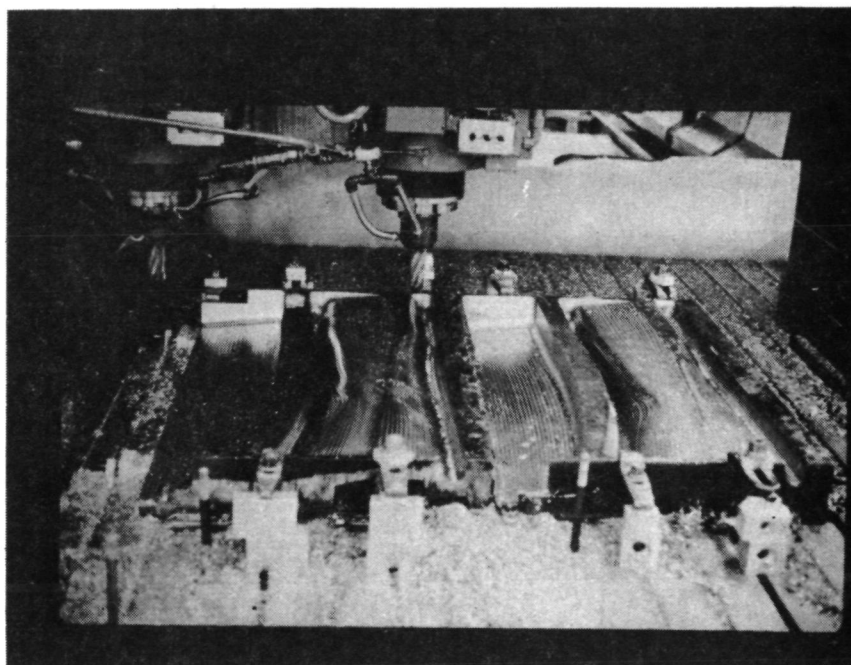


Figure 3.- Milling of blade surface tools.

# THE DESIGN AND IMPLEMENTATION OF CRT DISPLAYS IN

## THE TCV REAL-TIME SIMULATION

John B. Leavitt, Syed I. Tariq  
Electronic Associates, Inc.  
and  
George G. Steinmetz  
NASA Langley Research Center

### INTRODUCTION

NASA's Terminal Configured Vehicle (TCV) program is an advanced technology development **activity** focused on conventional transport aircraft that will operate routinely in reduced weather minima in future high-density terminal areas equipped with new landing systems, navigational aids, and increased air traffic control automation. The broad objectives of the program are to evaluate new concepts in air-borne systems (avionics and air vehicle) and operational flight procedures for reducing weather minima, increasing air traffic controller productivity and airport and airway capacity, saving fuel, and reducing noise by more efficient terminal techniques.

A Boeing 737-100 series aircraft was procured with a second flight deck and several computers installed in the passenger cabin. The aircraft is designed to be normally flown from the forward flight deck using the primary control systems and from the second (aft) flight deck using **digital fly-by-wire control techniques**. The aircraft can be flown under simulated category III conditions (zero visibility, zero ceiling) safely through the monitoring and take over capabilities of the forward flight deck crew. Included in the aircraft are two sets of computer driven CRT displays configured as the primary flight instrumentation indicating vertical and horizontal situation.

### THE TCV SIMULATION

One of the elements in support of the TCV program is a sophisticated simulation system. The simulation has been developed to duplicate the operation of the aft flight deck

in the 737 aircraft. A nearly identical cab was purchased from Boeing as part of the simulation system. The simulator facility can be used to develop and evaluate concepts of advanced flight procedures, provide pilot familiarization with advanced piloting techniques and provide a mechanism through which interested parties can see and evaluate research activities.

The simulation facility consists of an aft flight deck simulator, shown in figure 1, a CDC 6600 real-time simulation system, and an Adage Graphics Terminal. The simulator is equipped with realistic flight instrumentation including basic cockpit instruments, Automatic Guidance and Control panel (AGCS), Navigation Control and Display Unit (NCDU), Electronic Attitude and Direction Indicator (EADI), Electronic Horizontal Situation Indicator (EHSI), and panel mounted controllers, as shown in figure 2.

The simulation software includes the basic 737 nonlinear real-time simulation with the additions of landing gear dynamics, gust and wind models, nonlinear actuator models, and ILS and MLS (Instrument Landing System and Microwave Landing System) ground systems. Automatic flight control and navigation control functions have also been simulated and include control wheel steering, autoland with decrab and rollout capability, and navigation guidance and control. Bulk data storage for geographic map data, display computations, and NCDU computations on the CDC 6600 combined with the display generation capability of the Adage Graphics Terminal will present to the simulator pilot displays similar to those used on the aircraft with equivalent man-machine interaction capabilities. The two displays form the primary flight instrumentation for the pilot. The displays are dynamic images depicting critical flight information thus providing the pilot with a view of the flight situation.

The display equipment in the aircraft was tailored specifically for the TCV program and utilizes both raster and stroke drawing techniques. Although powerful, the hardwired and microprogrammed nature of the equipment makes it less flexible than desired for use in a laboratory environment. To provide the flexibility needed for the research program, an Adage system was chosen to generate the simulation displays. The Adage machine is very flexible from a programming standpoint and permits research in display formats to be realistically interwoven into the research schedule.

The Adage Graphics Terminal is a high speed interactive refresh-type graphics system. As configured at NASA Langley, it has 32K of core memory, a 30-bit CPU and four independently addressable CRT units. Hardware is provided to effect coordinate transformations. The Adage system is a completely self contained computer system with a powerful disk operating system, text editor and FORTRAN compiler. To support the cockpit displays, two remote CRT units were wired in parallel with two of the Adage scopes. In addition, three video scan conversion units are attached to the system, providing the capability for image transmission through NASA's extensive video distribution network.

### THE EADI DISPLAY

The EADI display, shown in figure 3, is the most primary display for the pilot. The EADI is a head-down type display for attitude, vertical situation, flight director commands and control information. The data presented can vary under control of the EADI mode control panel, depicted in figure 4. The EADI contains a number of dynamic elements which are updated at the rate of 16 times a second. The entire image is refreshed at 40 times a second. The EADI has a viewable area of 17.8 cm by 14.0 cm (7 inches by 5.5 inches). The EADI provides the pilot with information useful in vertical and horizontal flight maneuvers. For example, in a typical landing situation, the following information can be presented on the display:

1. pitch and bank angle information
2. altitude
3. deviation from ILS beam
4. a true perspective runway image
5. flight path and flight path acceleration
6. deviation from required heading.

The structure of the EADI display is defined for the most part by the Adage program and is modified dynamically by displacement and rotation values derived from data sent by the main simulation routines in the CDC 6600. A special case is the perspective runway, which is transmitted as an actual image list suitable for display without further processing.

## THE EHSI DISPLAY

The EHSI, shown in figure 5, is designed to provide the pilot with sufficient information for him to perform all horizontal and time-controlled flight maneuvers. The display replaces the electromechanical Horizontal Situation Indicator (HSI). It depicts that portion of the horizontal navigation and guidance information that lends itself to graphic representation and is of enough importance to warrant primary panel space. The EHSI has a viewable area 17.8 cm by 14.0 cm (7 inches by 5.5 inches). Data presented on the display are processed digitally in the CDC 6600 and the Adage. The display is updated 16 times a second. Image refresh is at 40 frames a second. These rates are sufficient for smooth dynamic flow at rates typically encountered in flight maneuvers. The image is controlled by the EHSI mode control panel shown in figure 6. The salient features depicted on the EHSI include:

1. aircraft horizontal position relative to a map showing flight path, airports, navaids, waypoints, boundary lines, and other points of interest
2. a curved trend indicator showing an extrapolated aircraft position 30, 60 and 90 seconds ahead
3. digital readouts for magnetic heading and ground speed
4. textual information indicating aircraft guidance mode and map scale
5. time guidance indicators.

## DISPLAY IMPLEMENTATION

The basic simulation equations of motion and control systems are designed for an iteration rate of 32 times a second as provided by NASA's CDC real-time computer systems. The I/O structure of the CDC 6600 and the real-time operating system support bi-directional data transfers once every frame (each frame is 1/32 of a second long). Due to the heavy I/O requirements of the simulation routines and the necessity to maintain this fixed frame rate, the time made available for 6600 to Adage transfers allows only 1200 bits to be transmitted each frame. The amount of data required to support the Adage displays far exceeds this number. A mechanism for accumulating display data and transmitting it in small blocks was developed. Fortunately, the displays need not be fully synchronized with the control programs since small delays

and inaccuracies are not discernable at a visual level by the pilot.

The data required by the Adage are logically divided into two major categories: real-time data and background data. The real-time data is characterized by the rate at which it is processed by the Adage display routines to maintain an illusion of continuous motion. The background image data are more static, being calculated over a period of several seconds.

To handle the influx of varied data, an interface package has been developed that provides the flexibility required. On the Adage, several double buffered input areas are defined to accommodate the transmitted blocks of each category. Each 1200 bit transmission may contain several data blocks of variable length headed by a type and length indicator. As data are generated and transmitted to the Adage, a "swap" code may be inserted to indicate that the data of a particular type has been fully transmitted. This causes the completed buffer load to be made available to the Adage display routines.

In this fashion, an asynchronous buffering technique was implemented that frees the CDC programmer from the fixed frame rate and fixed data length output restrictions imposed by the system. In addition, the ability to tag outgoing data as being destined for a particular buffer permits the CDC programmer to send actual image lists, if low update rates are permitted, or values to be applied dynamically to images already in Adage memory. Included as part of the simulation software are generalized Adage image support subroutines providing the CDC programmer with the capability to display fairly static images without becoming conversant with a complex stand-alone graphics system. These routines provide for data conversion, 2D and 3D clipping and perspective, and the generation of Adage format image lists. However, if the full power of the Adage is to be fully utilized, transmission of dynamic image modifying values and an appropriate Adage program are required. In the case of the TCV simulation, a hybrid approach was taken. Rapidly transmitted small blocks of data were used to modify the EADI and EHSDI dynamic displays, while the map vector and character data were accumulated slowly.

The key advantage to this approach is that it permits the simulation programmer to exploit the capabilities of each computer system to the fullest extent. In the case of the TCV simulation, it permitted the simultaneous display of

two complex images using a single graphics system. Additionally, this approach enabled programmers with a minimum of graphics experience to program a relatively large part of the display processing task.

## CONCLUSION

In many computing environments, the use of computer graphics is becoming increasingly attractive. Despite advances in hardware and software, graphic techniques remain somewhat mysterious to many programmers. In developing this simulation, we have devised methods to multiply the benefits of using a mini-computer based, stand-alone graphics system with a properly designed interface to a large simulation computer. Primarily, the division of computational load based on the abilities of the respective machines can be effected. These methods minimize the training requirements of simulation programmers wishing to use dynamic graphics. From a personnel standpoint, having a distinct dichotomy between the two worlds of graphics and simulation, in conjunction with a interface package similiar to the one described, permits a small number of graphics oriented people to support a large simulation community. A system such as the Adage can provide a great deal of power and flexibility not only for dedicated simulation uses, but for engineering applications in general. The techniques used in the successful integration of a variety of systems in support of the EADI and EHSD displays helped make the TCV simulation a useful and practical research tool.



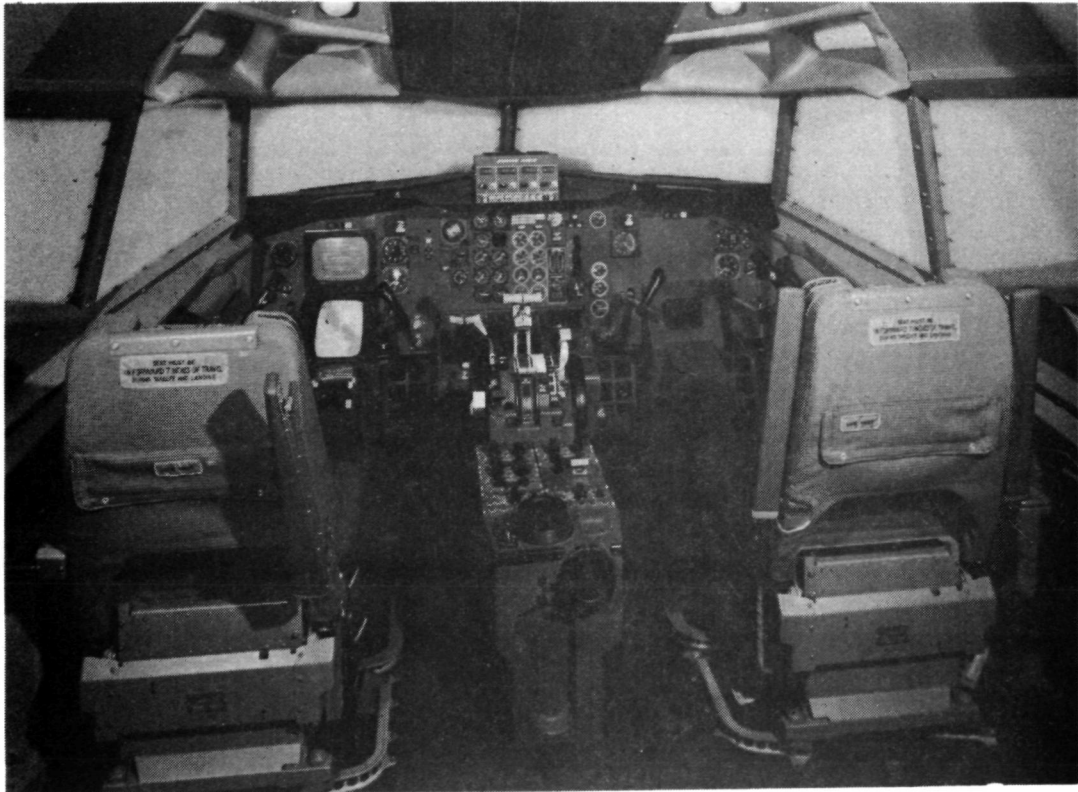


Figure 1 - Aft Flight Deck



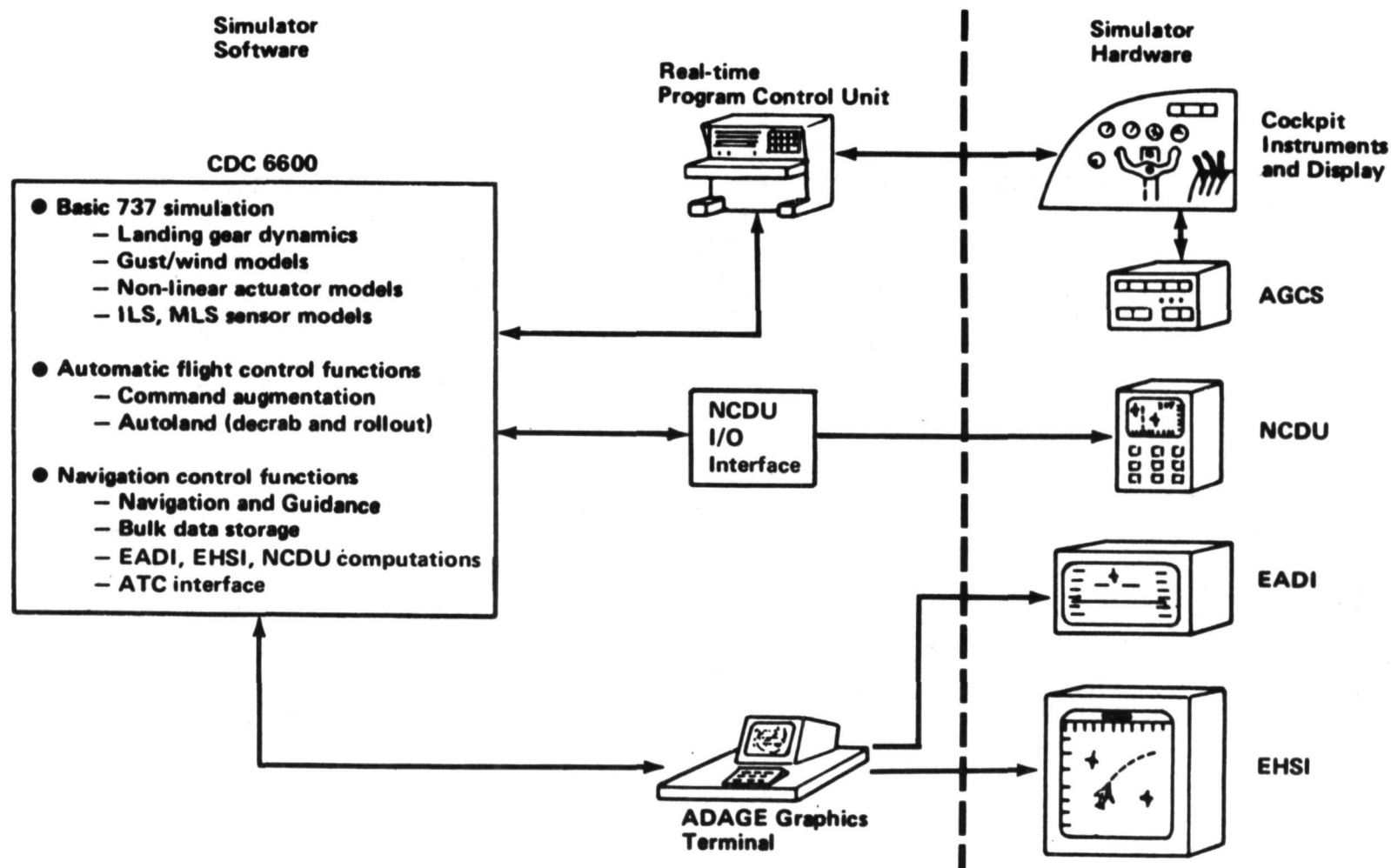


Figure 2 - Simulation System Block Diagram

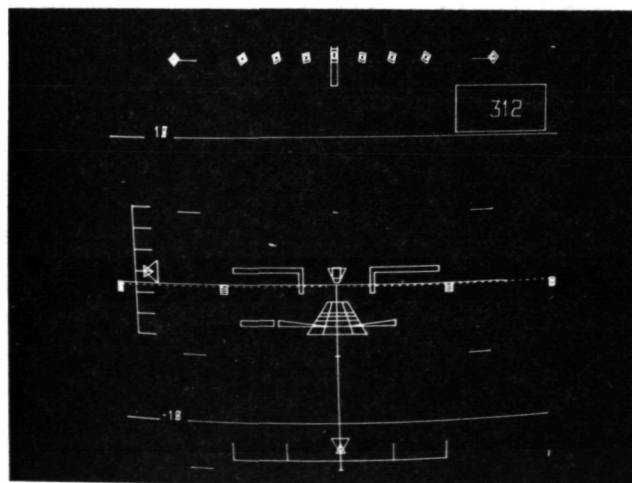
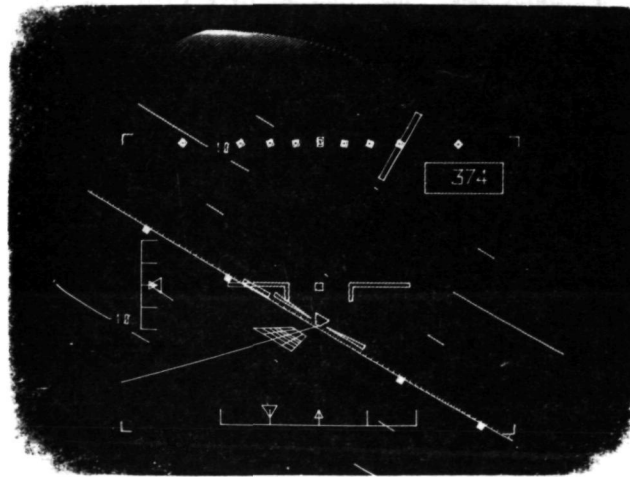
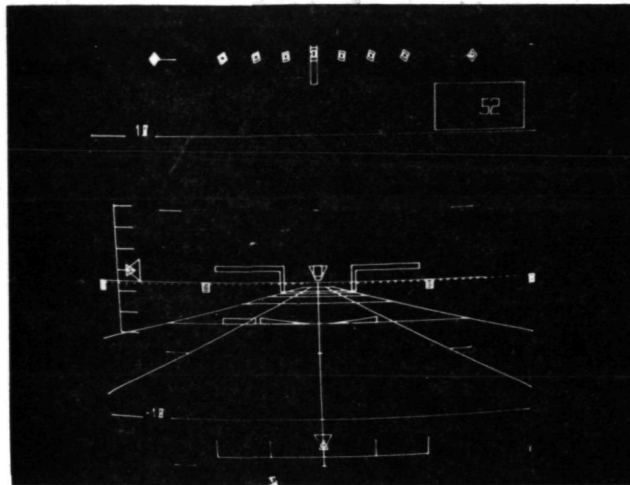


Figure 3 - EADI Displays

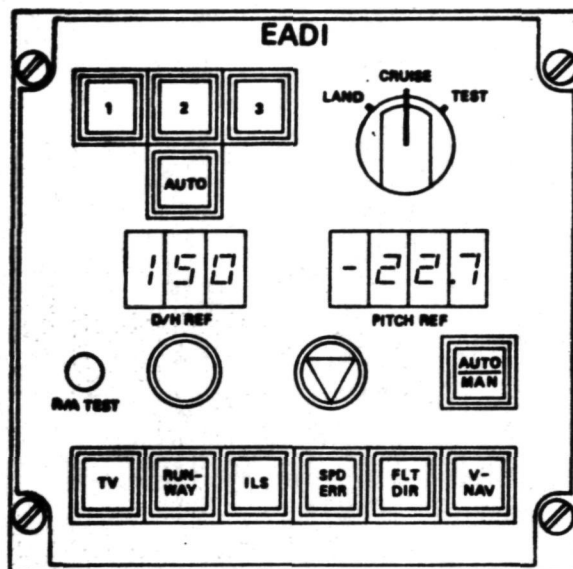


Figure 4 - EADI Mode Control Panel

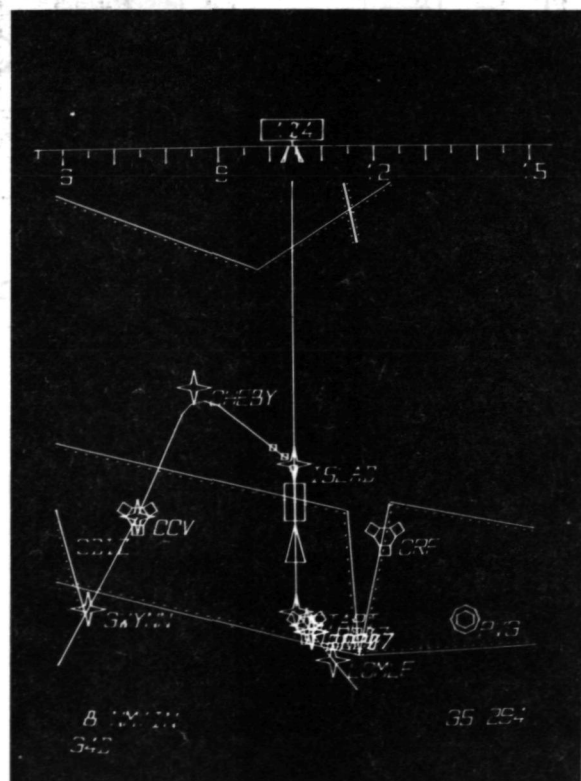
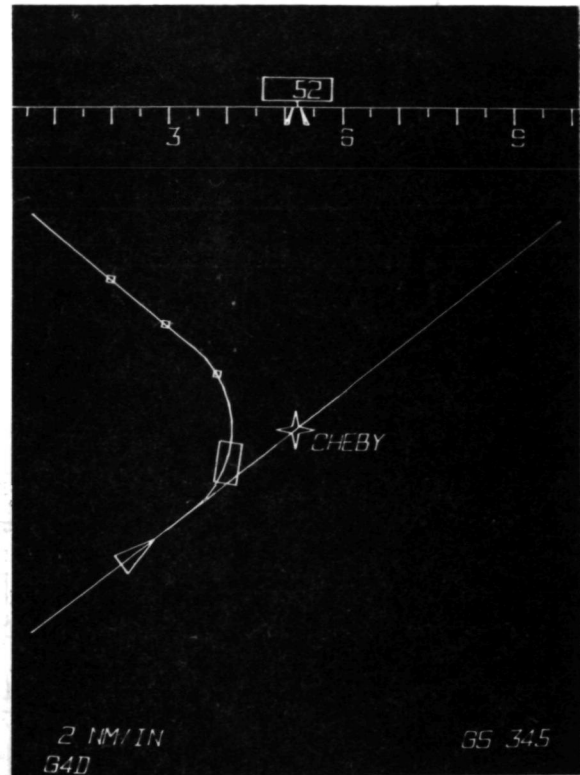
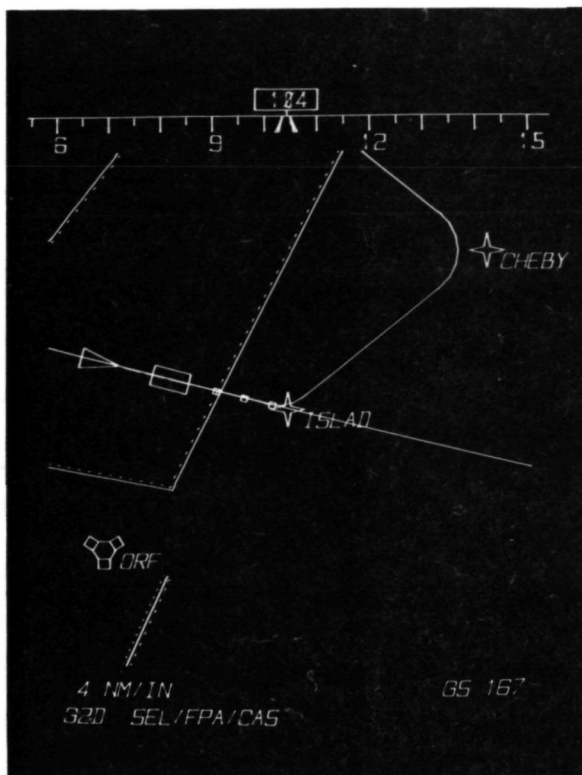


Figure 5 - EHSI Displays

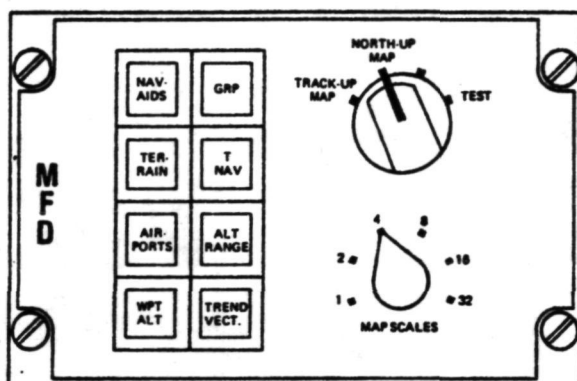


Figure 6 - EHSI Mode Control Panel

## PANEL DISCUSSION - PRACTICAL HARDWARE, SOFTWARE, AND SYSTEM CONSIDERATIONS

S. H. Chasen, Moderator, Lockheed-Georgia  
Carl Machover, Information Display, Inc.  
Richard Phillips, University of Michigan  
James Foley, University of North Carolina

### INTRODUCTION

The panel discussion was composed of two parts: (1) a brief presentation of each of the panelists on their area of interest, and (2) a general discussion among panel members and attendees. Included below are the issues raised by the panelists in their introductory remarks, followed by a transcription of the discussion.

Mr. Machover spoke on graphics system hardware and focused on the following issue:

How does one determine the proper choice of graphics hardware necessary to satisfy today's needs for a host-satellite or distributed processing system, and yet be cost-effective and expandable with the rapidly changing technology?

Dr. Phillips spoke on available graphics software and raised the following question:

Why is there an absence of good, practical graphics software; i.e., hidden-line processing, graph-producing routines, scan digitizing software, and interactive design data bases?

Dr. Foley spoke about the practical aspects of using satellite computers in support of graphics and raised the following issue:

What functions should be delegated to the satellite computer so as to speed picture generation and decrease required communication between host and satellite?

Mr. Chasen spoke on economic factors related to computer graphics and focused on the following issue:

How does one justify a computer graphics system to management?

## EDITED DISCUSSION

Chris Klomp, Boeing Company: I'd like to make a couple of comments regarding all four of you. Chase, the last slide you had on showed a decrease of cost per hour when you increase the number of consoles. Now, all graphics systems descriptions we have seen at this conference assumed that that particular application will save money. We have also listened to a talk by Professor Allan that said eventually we will leave the 'gee wiz' stage and will have to justify whatever we have done. Now, if one assumes that all these things come out correctly, as we all hope they will, then the logical conclusion is that graphics will be splattered all over the companies and then you are all of a sudden faced with a problem which is two or three magnitudes above the problems we have been laughing about here concerning response time and so forth. Conservative estimates in our company, for instance, show that we can easily reach the number of 500 graphic terminals in a relatively short time. So that gives us a problem that our graphic systems we have today will all of a sudden be toys and not any more represent the real scope of demand. The other comment I would like to make is that Mr. Machover mentioned all this beautiful hardware which will be available to us and I agree with him. His next comment was about the nonavailability of software to run that hardware. Well, we have managed in the last eight years to completely reverse the ratio and costs from software to hardware. It used to be the big problem to get a computer, and everybody was forgetting the software. Today, we are still talking like we ought to have all this beautiful hardware and then we have twenty people sitting in the background spending ten times as much money trying to keep the thing running. This, to me, represents the absolute necessity to maintain as much as possible hardware independent of pieces of software, in particular, when you are talking about graphics applications which are not on the level of Fortran callable subroutines and that sort of thing, but a normal noncomputer oriented user sitting at the console doing whatever they like to do in a broad engineering application environment.

Jim: I agree with everything you said.

Chase: Yes, Chris, this is hard to take issue with. I would like to add one comment - this idea of expanding into large numbers of terminals, even if you could cost justify it in the long range, say, you could show a 5 or 10 year payout. We have, I think, all of us recognized today a very great problem in capitalization and, therefore, even a good investment sometimes cannot be made if you don't have the capital to perform it.

Chris Klomp, Boeing: Well, I was really trying to make a very last point behind that, that's why I agreed with what you said about putting the intelligence as much as possible out to the satellite computers, because then you really can expand without having this great initial capital investment. You can do it step by step at that time.

Carl: I'm delighted with your forecast of 500 terminals at Boeing. I have a very narrow point of view of the world.

Chris Klomp, Boeing: Well, I'm not saying that we're going to have them next week.

Carl: I just cancelled my trip to Seattle. I would like to make an observation, because I think that I certainly agree with what you've said. I guess I sort of take issue that they're going to become toys unless you're using toys in the sense of becoming as common as a pocket calculator is now; in the sense that the device becomes available for almost anybody who wants to use it.

Chris Klomp, Boeing: No, no. I really didn't mean to imply that. What I wanted to imply was: if, in today's designer's system, we have a response problem with 20 terminals, the software or hardware configuration we are working on, and intent to expand possibly has failed in an initial design. If it cannot provide the flexibility to easily increase the number of users I am talking about. If it does not do that, then you really are talking about a toy at that time.

Carl: I agree. I would like to draw a slightly different analogy that I think fits with this. If we back up from computer graphics one step, I think most of us are used to using teletypes in a computer environment. It seems to me that we can view teletypes in the same relationship to the application problem that we can view computer graphics. Teletypes by themselves are a component, and a display by itself, maybe a component. You may be able to use teletypes interfaced to a large time-sharing system, just as you can use a display interfaced to a time-sharing system in which you are buying a service from a central service bureau. To a large extent, you have no knowledge of what's in that central bureau, as long as the system responds properly to you, it is satisfactory; if it doesn't, it's not satisfactory. On the other hand, you may not be in an environment in which the time sharing is suitable, then you buy literally a black box in which you may have squirrels on the inside. I'm exaggerating, of course, but the point is, I have a black box whose input/output port is a teletype and within that black box is the intelligence of whatever it is to solve this specific problem, and you may handle one or two of these and that's somewhat analogous to the question of distributive capacity. I may use the teletype in a completely different application that has nothing to do with computations, as I do with communication. And I guess what I'm really saying is that I think it is an unfair burden to put on computer graphics the task of making an efficient system design. The computer graphics is a tool, but the system design associated with it is the same kind of traditional problem that it was when you used a teletype as an input to the system.

Jim: It's utter foolishness to plug a black box satellite into a time-share computer system as though it were teletype and expect the user thereof to be satisfied. It's partly a question of what the expectations are, but



it's also partly a question of the rate at which the graphics user gets work done and presents processing load to the host computer, to the time-shared computer. At Michigan, I took some measurements - I don't remember the exact numbers anymore - but just for some simple interactive graphics applications, the user of the graphics terminal was presenting to the host computer a load from 5 to 10 times as great as the load that a typical teletype user was presenting. That says everything you plug your graphics terminal into your time-sharing computer, you should disconnect 5 to 10 teletypes if you're going to maintain a given level of service. Alternatively, it means when you plug in a graphics terminal, it's the same as plugging in 5 to 10 teletype users.

Chase: Okay, thank you. Any other questions, please.

W. A. Coles, British Aircraft Corporation: I would just like to make an observation on this business of the usefulness and the cost of graphics. In the case of electrical circuit diagrams, a number of manufacturers have come into being which use systems you can buy and can expand and this has not happened in the mechanical engineering drawing field. Normally, when you go to a conference you meet, you know, the vendors. I've actually not seen any vendors here and I just wonder, in fact, if, say the McDonnell Douglas or the Boeing system really is vastly superior for producing drawings. Why is it there is no commercial firm producing such a system?

Chase: As a Lockheedian, I guess I could answer for McDonnell and Boeing, but I best not. No, I think one observation is that they are terribly sophisticated systems. I think all the companies - Lockheed with its CADAM system, McDonnell with its CADD, and Boeing's development work - Boeing can speak for itself and McDonnell too if they wish to - but I think they're terribly complex systems that put all the bells and whistles together and have all the annotations, all the notation, everything such that you can store and retrieve in some systematic and orderly fashion. Also, the problem that the vendor, per se, other than these particular companies which may sell them, the vendor has to make a terribly great investment and a big decision as to what that market is going to be and he has to raise the capital to put in these many, many man-years of effort to produce these systems. So there are some logistical problems, but Carl can probably do better than that.

Carl: If you look at the field of computer-related drafting in general, there appear to be approximately 500 installed systems, manufactured by perhaps a half-dozen different companies: Computer Vision, Applicon, CALMA, IDI, Gerber; I'm sorry if I left anybody out. The statistics seem to show that about 85 percent of those installed systems at this point are involved in the electrical applications that are originally designed for LSI, printed circuit work, and some movement into the schematic area. The significant payoffs in those areas were because of return on investment, that there are companies that are using 20 to 30 of these stations because of the specialized software that has been developed in association with the hardware to do the productive kind of work. So, that says that of the 500 systems now installed, if 80 percent or

85 percent represent electrical applications, then 20 percent or about 100 are beginning to get into mechanical applications and that has been a real hassle. I think almost all who have been involved in the problems of 2-1/2 and 3D are not quite ready to throw up our hands, but we recognize that this is a severe problem. I don't know (unfortunately I was not here during the early sessions) but in addition to the large mechanical packages that have been developed by most of the aircraft companies, I think (is Lockheed the only one commercially marketing?) McDonnell has chosen not to, I don't know whether Boeing has, but Lockheed is commercially marketing their package. The other, and I'll put quotation marks around it, "classic 3D mechanical package" in the industry is the so-called Hanratty Package which was developed by Pat Hanratty and licensed to a number of companies that have come out with a variety of services and companies as a standard package. There have been industry problems in adapting that standard package to the individual pieces of equipment. If I were to make a forecast, it would seem to me that in the next couple of years, a dramatic and explosive growth area in computer graphics would be in the mechanical and associated areas having to do with common data bases, cartographics and things of that type. I think we're just beginning to see the major software breakthroughs now.

W. A. Coles, British Aircraft Corporation: Well, in very many cases, actually for the very large companies like Boeing, McDonnell, but for the smaller companies, it might be better for them to wait and buy this system when it arrives.

Carl: I'd like to address that one too. I think it goes back to the presentation that Chase made and that is that in most companies in the real world, the justification is economics. There are a lot of other reasons for getting into graphics, you know, you save time, you get better decisions and you can't do things other ways, but for most environments, the justification is economics. I would argue that if you could now economically justify graphics with the present software, and the present system, then the only thing you gain by waiting is to lose the money you could have saved now. In the long run, the money you could be saving now will not replace the greater productivity or the lower potential cost of the system coming down the pike. So the first task, it seems to me, is to decide whether, with what's available, you can cost justify graphics on whatever basis you choose to, and not to say, "Well, it's going to be cheaper next year, so I won't go at it this year." I really think that is not a correct decision.

Chase: To just extend what Carl said, it's a case of when do you cross that abscissa threshold and that is not necessarily next year, five years, or last year. It is the case of your particular mix of applications and your particular systems considerations that you have to take into account. Jim, do you want to say something?

Jim: Yes, I was going to say that everything I've heard people here saying reconfirms and strengthens my belief that before anyone nowadays starts a graphics system development, they should be doing it within the context of device independent graphics programming language so that 20 years from now,

when you want to get rid of those 2250's you're stuck with, you can do it.

Chase: Which one (programming language) would you recommend?

Jim: Later, later.

Chris Klomp, Boeing: I'd like to say something else since all these people are involved in graphics and are dealing with hardware, vendors, and so forth. One of the problems in trying to come up with the concept of distributive computing as it was described earlier is the commonality in communication protocols and the commitment by vendors to support some sort of commonality. Now, the normal answer is, there is no commonality. Well, I think we all agree that commonality in that area could save us an immense amount of money in software development, and it shouldn't be all that difficult to at least agree on two or three communication protocols. That way we have a choice and if we all start pushing in this area, I think the hardware industry would consider this earlier.

Jim: All I can say is that SIGGRAPH, the organization about which I spoke earlier, is trying very hard to establish some common base and, for instance, will be meeting in Minneapolis the latter part of October to work on that.

Chase: To what extent are they taking inputs from such people as Chris Klomp?

Jim: We welcome inputs. We try to publicize the activities mostly through SIGGRAPH vehicles. A lot of you don't know about SIGGRAPH, but we are very interested in hearing from users and from manufacturers who are willing to do things to establish some commonality.

Chase: In other words, what you are saying, if you want commonality, get involved? Get involved with the standards committee - the type of work that's going on.

Chris Klomp: Well, we are involved in the standards committee. It's more a moral commitment than anything else. We do agree with commonality if required and we do have a couple of committees trying to come up with a standard.

Chase: Do we have any other questions? Do you have a comment, Jim?

Jim: I was going to remark that for those of you who are interested in the area of graphics programming languages and perhaps are considering getting into graphics in a big way, SIGGRAPH and SIGPLAN will be sponsoring a workshop in April on computer graphics languages in Florida, and if you join SIGGRAPH you'll find out all about that.

Chase: What is it? \$7.00 and tax deductible?

Jim: \$8.00 and tax deductible.

Chase: Inflation has it too. Any other questions? I think the points that have been raised here are very relevant and very stimulating. I think we need to expound on these particular subjects; so feel free to speak up if you have any questions or any observations you'd like to make.

F. R. Zimmerman, Eustis Directorate, AMRDL: I'm quite new to the graphics business. We had a graphics terminal come in and a couple of us were told 'Here it is. Do something with it'. So the first thing we did was come to a conference. One of the things that has disturbed me for the past several years is that throughout a large part of the government we've been told, or management's been told, 'Here is a grand new thing, data base acquisition, graphics, graphics terminals; you can look at charts and find out what everybody in your organization is doing, how much they're making, and whether they're worth it. And of course, this started 10 years ago and none of this is true as far as the management is concerned. And I guess the question I have is how many people sitting at the table or sitting here in the audience have been involved in making management aware of what the problems are and other than justifying expensive equipment, making it clear to them what is really involved. For instance, the people I work for are quite well aware that there are nice graphics terminals, but they really are not aware that to make them work there has to be a giant data base and a giant computer behind it. And what I'm wondering is how much work is being done in the management organizations to clarify these problems.

Chase: What you are saying is one of the problems most of us that are involved in, in this business, try very hard to accomplish, but I think it sometimes becomes a matter of charter - who's on first and what's on second. What is the organizational setup by which you can properly have the opportunity first of all to clarify all these pros and cons and, quite frankly in some cases, to what extent is management capable of understanding what these tradeoffs are and, if they are not, then you have to make certain assumptions, certain simplifying assumptions in which they go off on a tangent and make certain decisions and you have to live with them. I don't know what realm of psychology or philosophy we're in here, and I don't want to think anyone has a panacea for handling that except to try very hard and keep motivated and proceed. All of us have great difficulty in trying to communicate the proper sense of balance to the right people. I say all of us, I think I can speak for some of us, but that's an observation - a personal opinion. Any others of you like to comment? Dick?

Dick: Generalized management information systems have their own problems, but let me just mention something that I feel is very true about graphics and managers. In a sense computer graphics is its own worst enemy. You would like to convince a manager of the extreme utility, and I'm talking about very low level practical applications, such as drawing the sorts of charts that you were suggesting, and some of the extraordinarily aesthetic

and beautiful things that graphics is capable of is the only thing that is seen by a manager and he immediately concludes that this is not for us; we can't make any use of this sort of thing. There is also a tendency, at least in my experience, on the part of management to feel that there is bound to be some extra cost associated, not with just the graphics hardware, but with using graphics equipment in a normal interrogation type of environment. You suggested data bases, so I use that term. And this is a very hard thing to overcome. It is generally true that it is not more expensive to do exactly the same activity from a graphics device. Of course, that depends upon the time-sharing system you are working with and how they charge for that service. But I think there's a very difficult education problem that has to be overcome before your management will accept graphics.

Chase: I might suggest that at a conference like this, maybe it's a little bit idealistic, but it would be very desirable if we could get some of our management to attend a conference like this and listen to what's going on. This might be one of the mediums of education that Dick speaks of. I know that IBM actually has workshops where they invite executives and I'm sure the other computer manufacturers must also do the same thing, invite executives to tell them essentially what's going on in these various fields, and they, therefore, make an attempt at this.

Carl: If I could make one other observation, I told Chase I talk too much, but I'll add to this. One of the things that seem to characterize engineering applications and characterize engineers is that they are used to thinking and viewing things pictorially and the shift from viewing things on paper and viewing charts that have been drawn by hand, to viewing them as being electronically generated and highly interactive is really not a major wrenching away of their traditional processes. Unfortunately, and it seems to me that one of the real problems that you're involved with from a management standpoint, is that management typically doesn't work with charts. Management typically works with numbers, and the older the management the more comfortable it is with numbers, the more comfortable it is with the numbers that are kept on a slip of paper and put in an inside coat pocket; so to some extent one of the things you're faced with is not only convincing them about the utility of this thing called graphics, but asking them to change the way in which they make decisions and that is not an easy task. It's an easier task to do it in engineering than it is in management. The second problem, it seems to me, that is fairly common is that it is fairly easy for us to identify common engineering practices. We can design, for example, an automatic drafting system because we have no standards and we know which way the dimensions ought to read and we know what paper sizes are available and where the three views are and there are all kinds of traditional practices that we can draw on; thus the material we're presenting to the engineers is not violently different in form or content than the way in which he has seen it before. And an engineer has essentially a common educational thread so that he is working with a vocabulary that is fairly common among a variety of industries. I would argue that management operates in a slightly different way. Each company's management information system

is different, its traditions are different, and its means of showing information is different. It, therefore, becomes difficult for you coming into the field to quickly take what in effect is a turn-key application that you can get in engineering and convert it into a management application because the management applications do not exist. So basically, it seems to me, the thing you really have to do, is to begin very carefully educating these management people in a dual mode, that whatever you give them in the way of graphics has all the numbers they have ever seen before at the same time. And very rapidly I think you'll find that they will look at the numbers, gradually realize that the numbers are being shown above in charted form, and make a very quick transition between the two. So I think you have to go through that phase.

Chase: Thank you, Carl. He is speaking as an Executive Vice-President of IDI.